SEQ 0001

.REM ?

IDENTIFICATION

PRODUCT CODE: AC-E968C-MC

PRODUCT NAME: CXRMBC0 RH11/RM03 DUAL PORT MOD

PRODUCT DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ABSTRACT

RMB IS AN IOMOD THAT EXERCISES RM03/RM02 DISK DRIVES ON AN
RH11 CONTROLLER. IT EXERCISES THE DRIVES BY DOING WRITES,
WRITE-CHECKS, READS, AND IN-CORE COMPARISONS. ALL ERRORS DETECTED
ARE REPORTED ON THE CONSOLE TTY.

2. REQUIREMENTS

HARDWARE: 1 TO 8 RM03/RM02'S WITH TWO RH11 CONTROLLERS

STORAGE:: RMB REQUIRES:
1. DECIMAL WORDS: 1793
2. OCTAL WORDS: 03401
3. OCTAL BYTES: 7002

3. PASS DEFINITION

ONE PASS OF THE RMB MODULE CONSISTS OF 300 CYCLES OF THE BASIC TEST
SEQUENCE. A-PORT DOES A WRITE, WRITE-CHECK, READ, DATA COMPARE
ON THE COURENT SECTOR (CALLED BLK THROUGHOUT PROGRAM) AND THEN
WRITES BLK 0 WITH BIT 4 SET IN THE FLAG WORD (THE FIRST WORD OF
BLOCK ZERO), AND THE SECOND WORD WITH THE CURRENT BLK.
A-PORT THEN SITS IN A LOOP, PERIODICALLY READING BLK 0 TO
SEE IF B-PORT HAS MODIFIED THE FLAG.
B-PORT SITS IN A LOOP WAITING FOR THE FLAG IN BLK ZERO TO
BE WRITTEN WITH BIT4 SET, WHEN IT IS B-PORT PICKS THE CURRENT
BLK FROM THIS DATA READ FROM BLK 0 AND GOES TO THAT SECTOR
AND READS THE DATA THAT A-PORT JUST WROTE. THEN B-PORT WRITES
THE DATA BACK AND WRITE CHECKS IT. NEXT IT RE-WRITES BLK 0
WITH BIT4 CLEARED AND WITH BIT2 SET, INDICATING TO A-PORT THAT HE IS DONE.
A-PORT AFTER READING BLK 0 AND SEEING THE FLAGS REVERSED, RE-READS
THE DATA WHICH IT HAD WRITTEN AND AGAIN DOES AN IN-CORE COMPARE.
THIS VERIFIES THE ABILITY OF A-PORT TO RE-READ DATA WHICH
IT HAD ORIGINALLY WRCTE BUT THAT HAD BEEN READ AND RE-WRITTEN
BY B-PORT. B-PORT DOES NOT DO ANY IN CORE COMPARISONS.

B PORT SHOULD NEVER ENCOUNTER A BAD BLK BECAUSE IT ONLY
USES BLK'S WHICH A-PORT HAS ALREADY SUCCESSFULLY USED.
THEREFORE, WHEN A-PORT GETS 4 ERRORS WHILE ATTEMPTING
A SEQUENCE ON ONE BLK, IT MOVES TO THE NEXT BLK, NEVER LETTING
B-PORT EVEN TRY THIS BAD BLK. IF B-PORT GETS 4 ERRORS ON
A BLK, THAT DRIVE IS DROPPED BECAUSE IT IS MOST LIKELY
FAULTY SINCE A-PORT WAS JUST ABLE TO SUCCESSFULLY USE THIS BLK.

LOCATION 206 THROUGH 244 CONTAIN ROOM FOR 16 BAD BLKS.
ON ERROR TYPEOUTS WHICH DUMP THE RH REGISTERS, THE LAST ITEM
TYPED IN THE TABLE IS THE CURRENT BLK NUMBER. ENTER THIS INTO
THE BAD BLK TABLE TO AVOID ERRORS FROM KNOWN MEDIA BAD SPOTS.
THIS TABLE ONLY MAKES SENSE ON THE A-PORT SIDE SINCE B-PORT

ALWAYS GETS ITS BLK ADDRESSES FROM A-PORT. IF YOU MODIFY
THE WRITE BUFFER SIZE, YOU MUST ADD SECTORS TO THE BAD BLK
TABLE TO AVOID THE ERRORS. FOR EXAMPLE, IF BLK 3474 IS BAD
AND YOU DOUBLE THE WRITE TRANSFER SIZE TO 1000 OCTAL BYTES
YOU MUST ADD BLK 3473 TO THE TABLE SO THE PROGRAM DOES NOT START
A TRANSFER THAT WILL EXTEND ON INTO THE KNOWN BAD BLK.

B-PORT MUST HAVE SR1 BIT 4 SET. ***** SR1 = 20 (OCTAL) *****
THIS MODULE IS A DUAL-PORT
MODULE TEST, AND ONLY USES THE FIRST 77777 BLKS
OF THE PACK. IF YOU WANT TO VERIFY THE COMPLETE MEDIUM, YOU
SHOULD RUN RMA MODULE, THE NORMAL SINGLE PORT TEST.

4. EXECUTION TIME

ONE PASS OF RMB RUNNING ALONE ON A PDP-11/70 TAKES APPROXIMATELY ONE MINUTE.

5. CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

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

REQUIRED PARAMETERS:

NONE

6. DEVICE/OPTION SETUP

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

THIS MODULE ALSO SUPPORTS RP04/5/6 ON THE SAME MASSBUS CONTROLLER.
HOWEVER,THIS MODULE IS NOT USED TO EXERCISE RP04/5/6 ALONE.

SEQ 0004

7. OPERATION OPTIONS

SR1 BIT2 SET(1):
COUNT DATA LATE ERRORS BUT DO NOT TYPE THEM OUT

SR1 BIT2 CLEAR(0):
TYPE OUT DATA LATE ERRORS AND COUNT THEM

SR1 BIT4 SET (1) ;-B-PORT PROGRAM *****SR1 = 20 (OCTAL) *****

SR1 BIT15 SET (1):
32 REGISTER OPTION ON RH70

SR1 BIT15 CLEAR (0):
22 REGISTER OPTION ON RH70

8. 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 RH11/RM03 REGISTERS IN THE FOLLOWING ORDER:

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

9. 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 7. FOR SR1 OPTIONS.

THE CONTROLLER SELECT SWITCH ON THE RM03 MUST BE IN THE A/B POSITION. THIS SWITCH IS ACTIVATED WHEN THE DRIVE IS CYCLED UP. IF SWITCH WAS NOT IN THIS POSITION WHEN DRIVE WAS POWERED UP, THE FOLLOWING STEPS MUST BE TAKEN. PLACE THE SWITCH IN THE A/B POSITION, DISABLE THE DRIVE, (USING THE DISABLE SWITCH ON THE DRIVE), THEN ENABLE THE DRIVE WITH THE SAME SWITCH. THIS WILL PUT THE RM03 IN THE DUAL PORT MODE.

10. BAD SPOT FILE

TOTAL 16 BAD BLOCKS CAN BE RETRIEVED FROM PORT A.
THE TABLE LABELED "BADSPT" IS SET UP TO RETRIEVE ALL BAD SPOT FILES FROM ALL DRIVES ASSIGNED IN THE BIT MAP "DVIDI".
(CYLINDER 822, TRACK 4, SECTOR 0 - MANUFACTUR BAD SPOT FILE)
(CYLINDER 822, TRACK 4, SECTOR 12 - USER BAD SPOT FILE)
LOCATION 1722 MUST BE SET TO 10 TO SEARCH USER BAD SPOT FILE

```
000000'          IOMODX  <RMBC  >,176700,254,5,0,0,300,145,RBUF,256.,256.
000000'          MODULE  150000,RMBC  ,176700,254,5,0,0,300,145,RBUF,256.,256.
                 .TITLE  RMBC DEC/X11 SYSTEM EXERCISER MODULE
                ;       DDXCOM  VERSION  6      23-MAY-78
                                .LIST   BIN
                                ;*****************************************************************
000000'                         BEGIN:
000600' 046522  041502     040  MODNAM: .ASCII  /RMBC / ;MODULE NAME.
000005'     066                 XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 176700                  ADDR:   176700+0                ;1ST DEVICE ADDR.
000010' 000 254                 VECTOR: 254+0                   ;1ST DEVICE VECTOR.
000012'     240                 BR1:    .BYTE   PRTY5+0         ;1ST BR LEVEL.
000013'     300                 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' 001604'                 INIT:   START                   ;MODULE START ADDR.
000032' 001252'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                       ;PASS COUNTER.
000036' 000300                  ICONT:  300                     ;# OF ITERATIONS PER PASS=300
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' 001236'                 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' 000145                  IDNUM:  145                     ;MODULE IDENTIFICATION NUMBER=145
000124' 020444'                 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                          .PEPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000252'                         MODSP:
                                ;*****************************************************************
000252' 000000                  BADSPT: 0                       ;BAD SPOT TABLE
000254' 000000                          0                       ;
000256' 000000                          0                       ;.
000260' 000000                          0                       ;
000262' 000000                          0                       ;BLK 0 IS NEVER CONSIDERED BAD
000264' 000000                          0                       ;IF YOURS IS, YOU CANNOT RUN THIS PROGRAM
000266' 000000                          0                       ;GO FIND ANOTHER PACK AND TRY AGAIN
000270' 000000                          0
000272' 000000                          0
000274' 000000                          0
000276' 000000                          0
000300' 000000                          0
000302' 000000                          0
000304' 000000                          0
000306' 000000                          0
000310' 000000                          0
000312' 000000  000000  000000  S:      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
000320' 000000  000000  000000
000326' 000000  000000  000000
000334' 000000  000000  000000
000342' 000000  000000  000000
000350' 000000  000000  000000
000356' 000000  000000  000000
000364' 000000                  ;DO NOT CHANGE THE ORDER OF THE NEXT 4 LOCATIONS
                                ;NEEDED FOR MAP22 ROUTINE
000366' 000000                  PA18:   0
000370' 000000                  XMEM:   0
000372' 000000                  PA22:   0
000374' 000000                  EA22:   0
000376' 000000                  BLK:    0
000400' 000000                  DSKADR: 0
000402' 000000                  CYL:    0
000404' 000000                  TRACK:  0
000406' 000000                  FLAG:   0
000410' 000000                  DVICE:  0
000412' 000000                  DRIVE:  0
000414' 000000                  UNITNO: 0
000416' 000000                  CNT:    0
000420' 000000                  CNT1:   0
```

```
000422' 000000          DLTCNT: 0
000424' 000000          FUNC:   0
000426' 000000          TIMER:  0
000430' 000000          ZERO:   0
000432' 000000          FERADR: 0
000434' 000000          CLK:    0
000436' 000000          TRY:    0
000440' 000000          MIXDV:  0       ;MIX DRIVE FLAG=-1 IF NOT RM03,RM02
000442' 000000          BADSEC: 0       ;HARDWARE DETECT BAD SECTOR
000444' 000400          RBUF:   .BLKW   256.

001444'                 TABLE:
001444' 000312'                 S
001446' 000314'                 S+2
001450' 000316'                 S+4
001452' 000320'                 S+6
001454' 000322'                 S+10
001456' 000324'                 S+12
001460' 000326'                 S+14
001462' 000330'                 S+16
001464' 000332'                 S+20
001466' 000334'                 S+22
001470' 000336'                 S+24
001472' 000340'                 S+26
001474' 000342'                 S+30
001476' 000344'                 S+32
001500' 000346'                 S+34
001502' 000350'                 S+36
001504' 000352'                 S+40
001506' 000354'                 S+42
001510' 000356'                 S+44
001512' 000360'                 S+46
001514' 000376'                 BLK
001516' 177777                  177777
001520' 000000          RMCS1:  0
001522' 000000          RMWC:   0
001524' 000000          RMBA:   0
001526' 000000          RMDA:   0
001530' 000000          RMCS2:  0
001532' 000000          RMDS:   0
001534' 000000          RMER1:  0
001536' 000000          RMAS:   0
001540' 000000          RMLA:   0
001542' 000000          RMDB:   0
001544' 000000          RMMR1:  0
001546' 000000          RMDT:   0
001550' 000000          RMSN:   0
001552' 000000          RMOF:   0
001554' 000000          RMDC:   0
001556' 000000          RMHR:   0
001560' 000000          RMMR2:  0
001562' 000000          RMEP2:  0
001564' 000000          RMEC1:  0
001566' 000000          RMEC2:  0
001570' 000000          RMBAE:  0
001572' 000000          RMCS3:  0
```



```
001574' 000432'         XFERAD: FERADR
001576' 000416'         XFERCT: CNT
001600' 000376'         XBLK:   BLK
001602' 177777                  177777
```

```
001604' 012767  000400  176302   START:  MOV     #256.,WDTO              ;256 WORDS TO MEM/ITERATION
001612' 012767  000400  176276           MOV     #256.,WDFR              ;256 WORDS FROM MEM/ITERATION
001620' 012767  000003  176272           MOV     #3,INTR                 ;3 INTERRUPTS/ITERATION
001626' 016767  176162  176554           MOV     DVID1,DVICE             ;GET DRIVES TO TEST
001634' 016706  176172                   MOV     SPOINT,R6               ;INIATE STACK POINT
001640' 012767  000001  176530           MOV     #1,BLK                  ;START AT BLOCK(SECTOR) 1
001646' 012767  000001  176532           MOV     #1,FLAG                 ;SET 1ST TIME FLAG BIT
001654' 012767  177777  176532           MOV     #-1,UNITNO              ;INITIATE UNIT NUMBER
001662' 004767  004102                   JSR     PC,SETUP
001666' 004767  002542                   JSR     PC,PICKDR               ;INITIATE UUT
001672' 000240                           NOP                             ;DUMMY ERROR RETURN
001674' 004767  004332                   JSR     PC,SETUP2               ;SETUP PSEL BIT
001700' 004767  003744                   JSR     PC,REZET                ;CLEAR THE RH
001704' 104415  000000' 000124'          GETPAS$,BEGIN, RBUFVA                   ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
001712' 005067  176500                   CLR     CNT                     ;RESET END-OF-PASS COUNTER
001716' 032767  000020  176072           BIT     #20,SR1                 ;R-PORT ?,
001724' 001161                           BNE     LOOP1                   ;THEN,DON'T HAVE TO RETRIEVE THE BAD SPOT FILE
001726' 012767  177777  176460           MOV     #-1,UNITNO              ;RESET UNIT NUMBER
001734' 012701  000252'                  MOV     #BADSPT,R1              ;TABLE ENTRY
001740' 012702  000020                   MOV     #16.,R2                 ;16 WORDS
001744' 005021                   1$:     CLR     (R1)+                   ;CLEAR THE TABLE
001746' 005302                           DEC     R2                      ;DECREMENT ONE WORD
001750' 001375                           BNE     1$                      ;BRANCH IF NOT DONE
001752' 004767  002456           RES2X:  JSR     PC,PICKDR               ;PICK UP THE FIRST ON LINE DRIVE
001756' 000526                           BR      10$                     ;BRANCH IF ALL DRIVES ARE ACCESSED
001760' 005067  176452                   CLR     TRY                     ;CLEAR RETRY COUNTER
001764' 012767  000770  176414           MOV     #770,FLAG               ;SPECIAL FLAG TAG FOR RETRIEVE BAD SPOT FILE
001772' 012767  001466  176402           MOV     #822.,CYL               ;CYL 822
002000' 112767  000004  176373           MOVB    #4,DSKADR+1             ;TRACK 4
002006' 112767  000000  176364           MOVB    #0,DSKADR               ;SEC 0 (SEC 12 USER)
002014' 004567  003440                   JSR     R5,READY                ;ACCESS THE DRIVE
002020' 000402                           BR      .+6                     ;BRANCH IF READY
002022' 004767  003174                   JSR     PC,NOTRDY               ;WAIT IF NOT READY
002026' 005767  176406                   TST     MIXDV                   ;EXIT,IF NOT RM03 OR RM02
002032' 100471                           BMI     9$                      ;EXIT
002034' 004567  001222                   JSR     R5,READ                 ;READ THE BAD SPOT FILE
002040' 012701  000444'                  MOV     #RBUF,R1                ;INPUT BUFFER
002044' 010102                           MOV     R1,R2                   ;END ADDRESS
002046' 062702  001000                   ADD     #512.,R2                ;LAST LOCATION OF THE INPUT BUFFER
002052' 062701  000010                   ADD     #10,R1                  ;BAD SOPT FILE STARTS FROM 5TH WORDS
002056' 012703  000252'                  MOV     #BADSPT,R3              ;BADSPT RELOCATABLE,TABLE ADDRESS
002062' 010304                           MOV     R3,R4                   ;LAST LOCATION OF THE TABLE
002064' 062704  000040                   ADD     #32.,R4                 ;16 WORDS
002070' 022711  000632           1$:     CMP     #410.,(R1)              ;BLOCK OVER CYL 410,THEN DON'T WORRY
002074' 101433                           BLOS    6$                      ;INCREMENT THE POINTER
002076' 005067  176274                   CLR     BLK                     ;INITIATE BLOCK COUNTER
002102' 011100                           MOV     (R1),R0                 ;CYLINDER ADDRESS
002104' 001405                           BEQ     3$                      ;BRANCH , IF CYL 0
002106' 062767  000240  176262   2$:     ADD     #32.*5,BLK              ;BLOCK # FOR ONE CYLINDER
002114' 005300                           DEC     R0                      ;DECREMENT TEMP COUNT
002116' 001373                           BNE     2$                      ;LOOP IF NOT ZERO
002120' 116100  000003           3$:     MOVB    3(R1),R0                ;TRACK NUMBER
002124' 001405                           BEQ     5$                      ;BRANCH IF TRACK 0
002126' 062767  000040  176242   4$:     ADD     #32.,BLK                ;ADJUST BLOCK FOR ONE TRACK
002134' 005300                           DEC     R0                      ;BRANCH IF NOT DONE
002136' 001373                           BNE     4$                      ;
```



```
002140' 116100  000002           5$:     MOVB    2(R1),R0                ;SECTOR NUMBER
002144' 060067  176226                   ADD     R0,BLK                  ;UPDATE SECTOR # FOR BLOCK COUNT
002150' 005713                           TST     (R3)                    ;ENTRY FOR THE TABLE ?
002152' 001016                           BNE     8$                      ;BRANCH IF NOT
002154' 016713  176216                   MOV     BLK,(R3)                ;LOAD INTO TABLE
002160' 062703  000002                   ADD     #2,R3                   ;UPDATE TABLE POINTER
002164' 062701  000004           6$:     ADD     #4,R1                   ;UPDATE THE BAD SPOT FILE POINTER
002170' 020102                   7$:     CMP     R1,R2                   ;END OF FILE
002172' 103011                           BHIS    9$                      ;BRANCH IF IT IS
002174' 020304                           CMP     R3,R4                   ;END OF TABLE ?
002176' 103734                           BLO     1$                      ;BRANCH IF NOT
002200' 104403  006766' 000000'          MSGN$,MF$15,BEGIN
002206' 000403                           BR      9$              ;DON'T DON ANYTHING
002210' 062703  000002           8$:     ADD     #2,R3           ;UPDTAE THE TABLE ENTRY
002214' 000765                           BR      7$              ;CHECK THE TABLE AND FILE ENTRY
002216' 000655                   9$:     BR      RES2X           ;TO NEXT DRIVE
002220' 012767  000001  176150   10$:    MOV     #1,BLK          ;RESET BLOCK NUMBER
002226' 012767  000001  176152           MOV     #1,FLAG                 ;RESTORE FIRST TIME FLAG
                                                                 ;COUNTINUE
002234' 000441                           BR       RSTRT1                 ;+ SUPPORT - DT03

002236' 005767  176156           RESTRT: TST     CNT1                    ;+       / SUPPORT
002242' 001005                           BNE     RSTRT1                  ;+      /  FOR
002244' 005767  176146                   TST     CNT                     ;+     /   DT03
002250' 001002                           BNE     RSTRT1                  ;+   /     BUS
002252' 000167  177326                   JMP     START                   ;+  /      SWITCH
002256'                          RSTRT1:                                 ;+/
002256' 104415  000000' 000124'          GETPAS$,BEGIN, RBUFVA                   ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
002264' 005067  176126                   CLR     CNT                     ;CLEAR THE LOCATE COUNTER
002270' 012767  177777  176116   LOOP1:  MOV     #-1,UNITNO              ;PRE-SET UNIT NUMBER
002276' 104414  000000'                  GWBUF$, BEGIN                   ;GET WRITE BUFFER INFORMATION
002302' 004767  002244                   JSR     PC,PICKBK               ;GO PICK A BLK
002306'                          LOOP2:
002306' 004767  002122           3$:     JSR     PC,PICKDR               ;GO PICK A DRIVE
002312' 000405                           BR      1$                      ;RETURNS HERE IF ALL DRIVES DONE
002314' 005067  176116                   CLR     TRY                     ;ELSE CLR RE TRY COUNT
002320' 004767  000034                   JSR     PC,CYCL                 ;GO DO A CYCLE ON THIS DRIVE
002324' 000770                           BR      LOOP2                   ;DO IT TO NEXT DRIVE

002326' 042767  000001  176052   1$:     BIC     #1,FLAG                 ;CLEAR FIRST TIME FLAG
002334' 005767  176050                   TST     DVICE                   ;ANYBODY LEFT TO CHECK?
002340' 001002                           BNE     2$                      ;BR IF YES
002342' 104410  000000'                  ENDS$,BEGIN                     ;
002346' 005267  176046           2$:     INC     CNT1                    ;+BUMP COUNTER FOR DT03
002352' 104413  000000'                  ENDIT$,BEGIN                    ;SIGNAL END OF ITERATION.
                                                                         ;MONITOR SHALL TEST END OF PASS
002356' 000744                           BR      LOOP1
```

```
                        ;B-PORT?
002360' 032767  000020  175430  CYCLE:  BIT     #20,SR1
002366' 001114                  BNE     CYCLER          ;BR IF SC, ELSE DO A-PORT
002370' 004567  003064          JSR     R5,READY        ;READY ?
002374' 000402                  BR      CYA             ;YES
002376' 004767  002620          JSR     PC,NOTRDY       ;NO
002402' 052767  000004  175776  CYA:    BIS     #4,FLAG         ;SET THE A-PORT FLAG
002410' 004767  001636          JSR     PC,BLKADR       ;CONVERT BLK TO DISK ADDR
002414' 004567  000452          JSR     R5,WRITE        ;GO WRITE A BLOCK
002420' 005767  176016          TST     BADSEC          ;FOUND BAD SPOT ?
002424' 001401                  BEQ     .+4             ;BRANCH IF NOT
002426' 000207                  RTS     PC              ;OTHERWISE EXIT
002430' 004567  000532          JSR     R5,WRITCK       ;GO DO WRITE CHECK
002434' 004567  000622          JSR     R5,READ         ;GO READ A BLOCK
002440' 104412  000000' 000126' CDATA$,BEGIN,RBUFFA     ; REQUEST FOR MONITOR TO CHECK DATA
002446' 002450'                 .+2                     ; IF ERROR, CONTINUE

                        ;NOW UPDATE BLOCK 0 FOR B-PORT'S INFORMATION

002450' 004767  002154          JSR     PC,CLRRB        ;CLEAR THE READ BUFFER
002454' 016767  175726  175762  MOV     FLAG,RBUF       ;PUT INFO IN FIRST WORD
002462' 016767  175710  175756  MOV     BLK,RBUF+2      ;PUT CURRENT ADDR IN NXT
002470' 004567  001052          JSR     R5,WRT00        ;GO UPDATE BLOCK 0
002474' 012767  000000  175724  MOV     #0,TIMER        ;INIT WAIT LOOP
002502' 012777  000013  177010  1$:     MOV     #13,@RMCS1      ;GIVE THE DRIVE TO B
002510' 005767  175712  2$:     TST     TIMER
002514' 001006                  BNE     3$
002516' 004767  002126          JSR     PC,DROP
002522' 104403  000000' 006744' MSGN$,BEGIN,PORTHG      ;ASCII MESSAGE CALL WITH COMMON HEADER
002530' 000207                  RTS     PC
002532' 012700  000002  3$:     MOV     #2,R0
002536'                 4$:
002536' 104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002542' 104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
002546' 005300                  DEC     R0
002550' 001372                  BNE     4$
002552' 004567  002702          JSR     R5,READY        ;READY ?
002556' 000402                  BR      5$              ;YES
002560' 004767  002436          JSR     PC,NOTRDY       ;NO
002564' 004567  001052  5$:     JSR     R5,RD00         ;GO READ BLOCK 0
002570' 000240                  NOP                     ;CHANGE TO BR IF ONLY SINGLE PORT
002572' 032767  000002  175644  BIT     #2,RBUF         ;HAS B UPDATED BLK 0 YET?
002600' 001740                  BEQ     1$              ;BR BACK IF NOT
002602' 004567  000454          JSR     R5,READ         ;GO READ THE DATA B WROTE
002606' 104412  000000' 000126' CDATA$,BEGIN,RBUFFA     ; REQUEST FOR MONITOR TO CHECK DATA
002614' 002616'                 .+2                     ; IF ERROR, CONTINUE
002616' 000207                  RTS     PC
```



```
002620' 012767  000000  175600  CYCLER: MOV     #0,TIMER
002626' 004567  002626          JSR     R5,READY        ;READY ?
002632' 000402                  BR      CYB             ;YES
002634' 004767  002362          JSR     PC,NOTRDY       ;NO
002640' 004567  000776  CYR:    JSR     R5,RD00         ;GO SEE IF A HAS DONE YET
002644' 032767  000001  175534  BIT     #1,FLAG         ;IS THIS THE FIRST TIME SINCE START?
002652' 001410                  BEQ     1$              ;BR IF NO, SKIP THIS CHECK
002654' 032767  000004  175562  BIT     #4,RBUF         ;WAS A WRITTEN THIS BLOCK?
002662' 001410                  BEQ     2$              ;BR IF NO
002664' 032767  000001  175552  BIT     #1,RBUF         ;HAS A SET THE FIRST TIME FLAG?
002672' 001404                  BEQ     2$              ;BR IF NOT, MUST WAIT
002674' 032767  000002  175542  1$:     BIT     #2,RBUF         ;HAS A WRITTEN THIS BLOCK SINCE B DID?
002702' 001432                  BEQ     3$              ;BR IF SO, ELSE
002704' 012777  000013  176606  2$:     MOV     #13,@RMCS1      ;GIVE PORT TO A
002712' 012700  000002          MOV     #2,R0
002716'                 4$:
002716' 104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002722' 104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
002726' 005300                  DEC     R0
002730' 001372                  BNE     4$
002732' 005767  175470          DEC     TIMER
002736' 001006                  BNE     6$
002740' 004767  001704          JSR     PC,DROP
002744' 104403  000000' 006744' MSGN$,BEGIN,PORTHG      ;ASCII MESSAGE CALL WITH COMMON HEADER
002752' 000207                  RTS     PC
002754' 004567  002500  6$:     JSR     R5,READY        ;READY ?
002760' 000727                  BR      CYB             ;YES
002762' 004767  002234          JSR     PC,NOTRDY       ;NO
002766' 000724                  BR      CYB             ;TRY AGAIN

002770' 016767  175452  175400  3$:     MOV     RBUF+2,BLK      ;GET THE CURRENT BLK
002776' 004767  001256          JSR     PC,BLKADR       ;GENERATE DISK ADDR FROM IT
003002' 004567  000254          JSR     R5,READ         ;GO READ WHAT A WROTE
003006' 004567  000344          JSR     R5,WRITEB       ;GO WRITE IT BACK OUT
003012' 004567  000434          JSR     R5,WRITCB       ;GO WRITE CHECK IT
003016' 004767  001606          JSR     PC,CLRRB        ;CLEAR BUFFER
003022' 052767  000002  175356  BIS     #2,FLAG         ;SET BIT SAYING B'S DONE
003030' 016767  175352  175400  MOV     FLAG,RBUF       ;PUT INFO INTO RBUF
003036' 004567  000504          JSR     R5,WRT00        ;GO WRITE IT FOR A TO SEE
003042' 012777  000013  176450  MOV     #13,@RMCS1      ;GIVE DRIVE TO A
003050' 012700  000002          MOV     #2,R0
003054'                 5$:
003054' 104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
003060' 104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
003064' 005300                  DEC     R0
003066' 001372                  BNE     5$
003070' 000207                  RTS     PC
```

```
                                        .MACRO  LINEUP  EABITS  ; LINE UP EA BITS FOR RHCS1
                                ;       LINEUP  EABITS          ; LINE UP EA BITS FOR RHCS1
                                        MOV     EABITS,R0       ; GET EXTENDED MEMORY BITS
                                        ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
                                        ASL     R0              ; TO LINE UP WITH RHCS1
                                        ASL     R0              ;
                                        ASL     R0              ;
                                        MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
                                        .ENDM   LINEUP

003072' 012767  000161' 175324  WRITE:  MOV     #161,FUNC       ; LOAD WRITE FUNCTION
003100' 012767  003072' 175324          MOV     #WRITE,FERADR   ;SAVE WHERE WE WERE
003106' 016746  175030                  MOV     WBUFSZ,-(SP)    ;GET WRITE SIZE
003112' 005416                          NEG     (SP)            ;NEGATE IT
003114' 012677  176402                  MOV     (SP)+,@RMWC     ; LOAD WORD COUNT
003120' 016777  175010  176376          MOV     WBUFPA,@RMBA    ; LOAD BUFFER ADDRESS
003126' 016777  175246  176372          MOV     DSKADR,@RMDA    ; LOAD DISK ADDRESS
003134' 016777  175242  176412          MOV     CYL,@RMDC       ; LOAD CYLINDER ADDRESS
                                ;       LINEUP  WBUFEA          ; LINE UP EA BITS FOR RHCS1
003142' 016700  174770                  MOV     WBUFEA,R0       ; GET EXTENDED MEMORY BITS
003146' 006300                          ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
003150' 006300                          ASL     R0              ; TO LINE UP WITH RHCS1
003152' 006300                          ASL     R0              ;
003154' 006300                          ASL     R0              ;
003156' 010067  175206                  MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
003162' 000167  000556                  JMP     GO              ; CONTINUE
003166' 012767  000151  175230  WRITCK: MOV     #151,FUNC       ; LOAD WRITE-CHECK FUNCTION
003174' 012767  003166' 175230          MOV     #WRITCK,FERADR  ;SAVE WHERE WE WERE
003202' 016746  174734                  MOV     WBUFSZ,-(SP)    ;GET WRITE SIZE
003206' 005416                          NEG     (SP)            ;NEGATE IT
003210' 012677  176306                  MOV     (SP)+,@RMWC     ; LOAD WORD COUNT
003214' 016777  174714  176302          MOV     WBUFPA,@RMBA    ; LOAD BUFFER ADDRESS
003222' 016777  175152  176276          MOV     DSKADR,@RMDA    ; LOAD DISK ADDRESS
003230' 016777  175148  176316          MOV     CYL,@RMDC       ; LOAD CYLINDER ADDRESS
                                ;       LINEUP  WBUFEA          ; LINE UP EA BITS FOR RHCS1
003236' 016700  174674                  MOV     WBUFEA,R0       ; GET EXTENDED MEMORY BITS
003242' 006300                          ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
003244' 006300                          ASL     R0              ; TO LINE UP WITH RHCS1
003246' 006300                          ASL     R0              ;
003250' 006300                          ASL     R0              ;
003252' 010067  175112                  MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
003256' 000167  000462                  JMP     GO              ; CONTINUE
003262' 012767  000171  175134  READ:   MOV     #171,FUNC       ; LOAD READ FUNCTION
003270' 012767  003262' 175134          MOV     #READ,FERADR    ;SAVE WHERE WE WERE
003276' 016746  174638                  MOV     RBUFSZ,-(SP)    ;GET READ SIZE
003302' 005416                          NEG     (SP)            ;NEGATE IT
003304' 012677  176212                  MOV     (SP)+,@RMWC     ; LOAD WORD CCUNT
003310' 016777  174612  176206          MOV     RBUFPA,@RMBA    ; LOAD BUFFER ADDRESS
003316' 016777  175056  176252          MOV     DSKADR,@RMDA    ; LOAD DISK ADDRESS
003324' 016777  175052  176222          MOV     CYL,@RMDC       ; LOAD CYLINDER ADDRESS
                                ;       LINEUP  RBUFEA          ; LINE UP EA BITS FOR RHCS1
003332' 016700  174572                  MOV     RBUFEA,R0       ; GET EXTENDED MEMORY BITS
003336' 006300                          ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
003340' 006300                          ASL     R0              ; TO LINE UP WITH RHCS1
003342' 006300                          ASL     R0              ;
```



```
003344' 006300                          ASL     R0              ;
003346' 010067  175016                  MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
003352' 000167  000366                  JMP     GO              ; CONTINUE
003356' 012767  000161  175040  WRITEB: MOV     #161,FUNC       ; LOAD WRITE FUNCTION
003364' 012767  003356' 175040          MOV     #WRITEB,FERADR  ;SAVE WHERE WE WERE
003372' 016746  174534                  MOV     RBUFSZ,-(SP)    ;GET WRITE SIZE
003376' 005416                          NEG     (SP)            ;NEGATE IT
003400' 012677  176116                  MOV     (SP)+,@RMWC     ; LOAD WORD COUNT
003404' 016777  174516  176112          MOV     RBUFPA,@RMBA    ; LOAD BUFFER ADDRESS
003412' 016777  174762  176106          MOV     DSKADR,@RMDA    ; LOAD DISK ADDRESS
003420' 016777  174756  176126          MOV     CYL,@RMDC       ; LOAD CYLINDER ADDRESS
                                ;       LINEUP  RBUFEA          ; LINE UP EA BITS FOR RHCS1
003426' 016700  174476                  MOV     RBUFEA,R0       ; GET EXTENDED MEMORY BITS
003432' 006300                          ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
003434' 006300                          ASL     R0              ; TO LINE UP WITH RHCS1
003436' 006300                          ASL     R0              ;
003440' 006300                          ASL     R0              ;
003442' 010067  174722                  MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
003446' 000167  000272                  JMP     GO              ; CONTINUE
003452' 012767  000151  174744  WRITCB: MOV     #151,FUNC       ; LOAD WRITE-CHECK FUNCTION
003460' 012767  003452' 174744          MOV     #WRITCB,FERADR  ;SAVE WHERE WE WERE
003466' 016746  174440                  MOV     RBUFSZ,-(SP)    ;GET WRITE SIZE
003472' 005416                          NEG     (SP)            ;NEGATE IT
003474' 012677  176022                  MOV     (SP)+,@RMWC     ; LOAD WORD COUNT
003500' 016777  174422  176016          MOV     RBUFPA,@RMBA    ; LOAD BUFFER ADDRESS
003506' 016777  174666  176112          MOV     DSKADR,@RMDA    ; LOAD DISK ADDRESS
003514' 016777  174662  176032          MOV     CYL,@RMDC       ; LOAD CYLINDER ADDRESS
                                ;       LINEUP  RBUFEA          ; LINE UP EA BITS FOR RHCS1
003522' 016700  174402                  MOV     RBUFEA,R0       ; GET EXTENDED MEMORY BITS
003526' 006300                          ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
003530' 006300                          ASL     R0              ; TO LINE UP WITH RHCS1
003532' 006300                          ASL     R0              ;
003534' 006300                          ASL     R0              ;
003536' 010067  174626                  MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
003542' 000167  000176                  JMP     GO              ; CONTINUE
003546' 012767  000161  174650  WRT00:  MOV     #161,FUNC       ; LOAD WRITE FUNCTION
003554' 012767  003546' 174650          MOV     #WRT00,FERADR   ;SAVE WHERE WE WERE
003562' 016746  174344          WRT00R: MOV     RBUFSZ,-(SP)    ;GET WRITE SIZE
003566' 005416                          NEG     (SP)            ;NEGATE IT
003570' 012677  175726                  MOV     (SP)+,@RMWC     ; LOAD WORD COUNT
003574' 016777  174326  175722          MOV     RBUFPA,@RMBA    ; LOAD BUFFER ADDRESS
003602' 012777  000000  175716          MOV     #0,@RMDA        ; LOAD DISK ADDRESS
003610' 012777  000000  175736          MOV     #0,@RMDC        ; LOAD CYLINDER ADDRESS
                                ;       LINEUP  RBUFEA          ; LINE UP EA BITS FOR RHCS1
003616' 016700  174306                  MOV     RBUFEA,R0       ; GET EXTENDED MEMORY BITS
003622' 006300                          ASL     R0              ; SHIFT 4 PLACES TO THE LEFT
003624' 006300                          ASL     R0              ; TO LINE UP WITH RHCS1
003626' 006300                          ASL     R0              ;
003630' 006300                          ASL     R0              ;
003632' 010067  174532                  MOV     R0,XMEM         ; SAVE THE SHIFTED BITS
003636' 000167  000102                  JMP     GO              ; CONTINUE
003642' 012767  000171  174554  RD00:   MOV     #171,FUNC       ;SET FUNCTION TO A READ
003650' 012767  003442' 174554          MOV     #RD00,FERADR    ;SAVE WHERE WE WERE
003656' 000741                          BR      WRT00           ;THEN USE WRT00 ROUTINE FOR THE READ

003660' 016777  174530  175642  CLEAR:  MOV     UNITNO,@RMCS2   ; LOAD UNIT ADDRESS
```

```
 680  003666' 012777  000011  175624          MOV     #11,@RMCS1      ; ISSUE A DRIVE CLEAR
 681  003674' 000240                          NOP                     ;WAIT
 682  003676' 000240                          NOP                     ;FOR DRIVE CLEAR TO FINISH
 683  003700' 012777  000023  175612          MOV     #23,@RMCS1      ;ISSUE A PACK ACK
 684  003706' 105777  175606          1$:     TSTB    @RMCS1
 685  003712' 100401                          BMI     2$
 686  003714' 000774                          BR      1$              ; NO, WAIT TILL DONE
 687  003716' 012746  175614          2$:     MOV     @RMAS,-(SP)     ;CLEAR AS BIT
 688  003722' 012677  175610                  MOV     (SP)+,@RMAS
 689  003726' 012777  040000  175564          MOV     #BIT14,@RMCS1   ; CLEAR ANY CONTROLLER ERRORS
 690  003734' 012777  010000  175610          MOV     #BIT12,@RMDP    ; SET BIT FOR 11 FORMAT
 691  003742' 000205                          RTS     R5              ; RETURN
 692
 693  003744' 016777  174444  175556  GO:     MOV     UNITNO,@RMCS2   ; LOAD UNIT SELECT
 694  003752' 032767  001000  174076          BIT     #ADDR22,RESI    ;11/70 MONITOR?
 695  003760' 001434                          BEQ     1$              ;NO
 696  003762' 017767  175536  174376          MOV     @RMBA,PA18      ;GET 18 BIT ADDR
 697  003770' 006267  174374                  ASR     XMEM            ;SHIFT EA BITS TO POSITION 4,5
 698  003774' 006267  174370                  ASR     XMEM
 699  004000' 006267  174364                  ASR     XMEM
 700  004004' 006267  174360                  ASR     XMEM
 701  004010' 104416  000000' 000366'        MAP22$, BEGIN,PA18              ; GET 22-BIT ADDR FROM 18-BIT ADDR
 702  004016' 016777  174350  175500          MOV     PA22,@RMBA      ;LOAD BA REG
 703  004024' 016777  174344  175536          MOV     EA22,@RMBAE     ;LOAD BAE REG
 704  004032' 042767  000034  174334          BIC     #34,EA22        ;CLEAR UNWANTED BITS
 705  004040' 000367  174330                  SWAB    EA22            ;LOAD INTO BITS 8,9
 706  004044' 016767  174324  174316          MOV     EA22,XMEM       ;LOAD XMEM TO SET INTO FUNCTION CODE
 707  004052' 056767  174312  174344  1$:     BIS     XMEM,FUNC       ; LOAD EXTENDED MEMORY BITS
 708  004060' 016797  174340  175432          MOV     FUNC,@RMCS1     ; EXECUTE THE FUNCTION
 709  004066' 104400  000000'                 EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
 710  004072' 010046                  NTRUPT: MOV     R0,-(SP)        ;SAVE R0
 711  004074' 016700  174314                  MOV     UNITNO,R0       ;DRIVE #
 712  004100' 116000  004542'                 MOVB    BITTAB(R0),R0   ;DRIVE INDICATOR
 713  004104' 017746  175426                  MOV     @RMAS,-(SP)     ;CLEAR CORESPONDING
 714  004110' 040016                          BIC     R0,(SP)         ;ATTENTION BIT
 715  004112' 012677  175420                  MOV     (R6)+,@RMAS     ;IN THE ATTENTION SUMMARY REGISTER
 716  004116' 012600                          MOV     (SP)+,R0        ;RESTORE R0
 717                                          ;-----------------------------------------------------------------------
 718  004120' 000004  000000' 004126          PIRQ$,BEGIN,1$          ; QUEUE UP TO CONTINUE AT 1$ AND RTI
 719                                          ;-----------------------------------------------------------------------
 720
 721  004126' 004567  000554          1$:     JSR     R5,ERRORS       ; GO CHK FOR ERRORS
 722  004132' 000401                          BR      2$              ;ERRORS DETECTED
 723  004134' 000205                          RTS     R5              ;OTHERWISE,  RETURN OK
 724
 725  004136' 105267  174274          2$:     INC     TRY             ;COUNT AN ERROR
 726  004142' 026767  174270  000004          CMP     TRY,#4          ;TOO MANY FOR THIS CYCLE?
 727  004150' 001035                          BNE     4$              ;BR IF NOT
 728  004152' 104403  000000' 006724'        MSGN$,BEGIN,EXCED       ;ASCII MESSAGE CALL WITH COMMON HEADER
 729  004160' 012605                          MOV     (SP)+,R5        ;FOR RETURN WE WONT DO
 730  004162' 032767  000020  173624          BIT     #20,SR1
 731  004170' 001411                          BEQ     3$
 732  004172' 004767  000452                  JSR     PC,DROP
 733  004176' 104403  000000' 006754'        MSGN$,BEGIN,DRP ;ASCII MESSAGE CALL WITH COMMON HEADER
 734  004204' 016706  173622                  MOV     SPOINT,R6
 735  004210' 000167  176072                  JMP     LOOP2
```



```
 736  004214' 032767  000770  174164  3$:     CMP     #770,FLAG       ;IN PROCESS OF RETRIVING BAD SPOT ?
 737  004222' 001002                          BNE     +6              ;BRANCH IF NOT
 738  004224' 000167  175522                  JMP     RES2X           ;IF SO, THEN BRANCH BACK
 739  004230' 004767  000316                  JSR     PC,PICKBK       ;TRY A DIFFERENT BLOCK
 740  004234' 005367  174154                  DEC     UNITNO          ;WANT TO RE-DO SAME DRIVE WERE ON
 741  004240' 000167  176042                  JMP     LOOP2           ;GO DO IT
 742  004244' 162705  000004          4$:     SUB     #4,R5           ;BUMP BACK RETURN FOR A RETRY
 743  004250' 000205                          RTS     R5
 744
 745                                  ;       -------------------------
 746
 747  004252' 005067  174126          BLKADR: CLR     TRACK           ;START ADDR AT ZERO
 748  004256' 005067  174116                  CLR     DSKADR
 749  004262' 005067  174114                  CLR     CYL
 750  004266' 005767  174146                  TST     MIXDV           ;MIXED DRIVE ?
 751  004272' 100430                          BMI     5$              ;BRANCH IF SO
 752  004274' 016700  174076                  MOV     BLK,R0          ;GET BLOCK NUMBER
 753  004300' 162700  000240          1$:     SUB     #32.*5.,R0              ;SUBTRACT A CYLINDER'S WORTH OF SECTORS
 754  004304' 100403                          BMI     2$              ;BR OUT IF WENT TOO FAR
 755  004306' 005267  174070                  INC     CYL             ;ELSE COUNT A CYLINDER
 756  004312' 000772                          BR      1$              ;AND DO IT AGAIN
 757  004314' 062700  000240          2$:     ADD     #32.*5.,R0              ;PUT BACK ONE
 758  004320' 162700  000040          3$:     SUB     #32.,R0         ;SUBTRACT A TRACK'S WORTH OF SECTORS
 759  004324' 100403                          BMI     4$              ;BRANCH IF WE WENT TO FAR
 760  004326' 105267  174052                  INCB    TRACK+1         ;ELSE COUNT ANOTHER TRACK
 761  004332' 000772                          BR      3$              ;AND DO IT AGAIN
 762  004334' 062700  000040          4$:     ADD     #32.,R0         ;PUT BACK ONE
 763  004340' 010067  174034                  MOV     R0,DSKADR       ;PUT IN SECTOR ADDRESS
 764  004344' 116767  174031  174027          MOVB    TRACK+1,DSKADR+1        ;PUT IN TRACK ADDR
 765  004352' 000427                          BR      10$             ;EXIT
 766  004354' 016700  174016          5$:     MOV     BLK,R0          ;LOAD THE BLOCK NUMBER
 767  004360' 162700  000640          6$:     SUB     #22.*19.,R0     ;22 SECTOR*19 TRACK
 768  004364' 100415                          BMI     7$              ;DONE IF NEG
 769  004366' 005267  174011                  INC     CYL             ;CYLINDER ADDRESS
 770  004372' 000772                          BR      6$              ;LOOPING BACK
 771  004374' 062700  000640          7$:     ADD     #22.*19.,R0     ;ADJUST ONE CYLINDER
 772  004400' 162700  000026          8$:     SUB     #22.,R0         ;FIND WHICH TRACK
 773  004404' 100403                          BMI     9$              ;EXIT IF DONE
 774  004406' 105267  173777                  INCB    TRACK+1         ;INCREMENT TRACK COUNT
 775  004412' 000772                          BR      8$              ;BRANCH IF NOT DONE
 776  004414' 062700  000026          9$:     ADD     #22.,R0         ;ADJUST ONE TRACK
 777  004420' 010067  173757                  MOV     R0,DSKADR       ;TRACK AND SECTOR ADDRESSES
 778  004424' 116767  173754  173747          MOVB    TRACK+1,DSKADR+1        ;
 779  004432' 000207                  10$:    RTS     PC
 780
 781  004434'                         PICKDR:
 782  004434' 005067  174000                  CLR     MIXDV           ;RESET  THE MIX DRIVE FLAG
 783  004440' 005267  173750          1$:     INC     UNITNO          ;POINT TO NEXT DRIVE
 784  004444' 026727  173744  000010          CMP     UNITNO,#8.              ;DONE LOOKING?
 785  004452' 002401                          BLT     2$              ;BR IF NO, ELSE
 786  004454' 000207                          RTS     PC
 787  004456' 016700  173732          2$:     MOV     UNITNO,R0       ;USE AS AN INDEX
 788  004462' 136067  004542' 173720          BITB    BITTAB(R0),DVICE        ;TEST THIS DEVICE?
 789  004470' 001001                          BNE     3$              ;BR IF YES, ELSE
 790  004472' 000763                          BR      1$
 791  004474' 062716  000002          3$:     ADD     #2,(SP)         ;BUMP FOR A GOOD RETURN
```

```
 792  004500' 004567 000754            JSR     R5,READY        ;SEE IF DRIVE IS READY
 793  004504' 000402                   BR      4$              ;BR IF IT WAS READY
 794  004506' 004767 000510            JSR     PC,NOTRDY       ;ELSE GO CLEAR IT AND CHECK AGAIN
 795  004512' 022777 024025 175026 4$: CMP     #24025,@RMDT    ;RM02 ?
 796  004520' 001407                   BEQ     5$              ;EXIT IF SO
 797  004522' 022777 024024 175016     CMP     #24024,@RMDT    ;RM03 ?
 798  004530' 001403                   BEQ     5$              ;BRANCH IF SO
 799  004532' 012767 177777 173700     MOV     #-1,MIXDV       ;OTHERWISE SET THE FLAG
 800  004540' 000207                   RTS     PC              ;RETURN WITH A UNITNO READY
 801  004542' 001001           5$:  BITTAB: 1001
 802  004544' 004004                   4004
 803  004546' 020020                   20020
 804  004550' 100100                   100100
 805
 806
 807  004552' 062767 000001 173616 PICKBK: ADD     #1,BLK          ;DO NEXT BLOCK(SECTOR)
 808  004560' 026727 173612 077777     CMP     BLK,#77777
 809  004566' 103403                   BLO     1$
 810  004570' 012767 000001 173600     MOV     #1,BLK          ;GO BACK TO BLOCK 1
 811  004576' 005767 173636        1$: TST     MIXDV           ;NOT RM03/RM02 ?
 812  004602' 100411                   BMI     3$              ;BRANCH IF NOT
 813  004604' 012700 000252'           MOV     #BADSPT,R0      ;GET BAD SPOT TABLE
 814  004610' 012701 000020            MOV     #16.,R1         ;LOOK FOR 16 ENTRIES
 815  004614' 026720 173556        2$: CMP     BLK,(R0)+       ;IS THIS A BAD BLK?
 816  004620' 001754                   BEQ     PICKBK          ;IF YES, GO PICK A NEW ONE
 817  004622' 005301                   DEC     R1              ;COUNT A TABLE LOOK-UP
 818  004624' 001373                   BNE     2$              ;BR BACK IF MORE TO GO
 819  004626' 000207               3$: RTS     PC
 820
 821  004630' 012700 000444'  CLRRB:   MOV     #RBUF,R0        ;CLEAR RBUF BUFFER
 822  004634' 012701 173272            MOV     RBUFSZ,R1       ;GET ITS ADDR AND SIZE
 823  004640' 005020           CLRCOM: CLR     (R0)+           ;CLEAR ANOTHER
 824  004642' 005301                   DEC     R1              ;COUNT ANOTHER
 825  004644' 001375                   BNE     CLRCOM          ;BR BACK TILL DONE
 826  004646' 000207                   RTS     PC
 827
 828  004650' 012701 000001  DROP:     MOV     #1,R1           ; INITIALIZE DROP PICKER
 829  004654' 016700 173534            MOV     UNITNO,R0       ; GET THE DRIVE NUMBER
 830  004660' 001403                   BEQ     2$              ; IF DRIVE 0 GO DROP IT
 831  004662' 006301               1$: ASL     R1              ; POINT TO NEXT DRIVE
 832  004664' 005300                   DEC     R0              ; IS THIS THE ONE ?
 833  004666' 001375                   BNE     1$              ; NO, LOOK AGAIN
 834  004670' 050167 173514        2$: BIC     R1,DVICE        ; DROP THE DRIVE
 835                                   ;*****************************************************************
 836                                                           ;CONVERT UNITNO TO ASCII AND
 837                                                           ;STORE AT ADR1
 838  004674' 1 4420  000000' 000414'  OTOA$,BEGIN,UNITNO,ADR1
 839  004702' A 4772'
 840                                   ;*****************************************************************
 841  004704' 000207                   RTS     PC              ; RETURN
 842                                ;  ----------------------
 843
 844  004706' 005067 173530  ERRORS:   CLR     BADSEC          ;
 845  004712' 005777 174002            TST     @RMCS1          ; ATTENTION OR ERROR ?
 846  004716' 100402                   BMI     1$              ;BR IF YES
 847  004720' 005725                   TST     (R5)+           ;BUMP FOR GOOD RETURN
```



```
 848  004722' 000205                   RTS     R5              ;AND RETURN
 849  004724' 005000           1$:     CLR     R0
 850  004726' 016701 174566            MOV     RMCS1,R1        ;ADDRESS OF RMCS1
 851  004732' 012160 000317'   22$:     MOV     (R1)+,S(R0)     ;READ AND STORE
 852  004736' 005720                   TST     (R0)+           ;ALL CONTROLLER AND RM03
 853  004740' 020760 000046            CMP     #46,R0          ;REGISTERS
 854  004744' 001372                   BNE     22$
 855  004746' 004767 000666            JSR     PC,ERSUB1       ; LOAD ERROR INFORMATION
 856  004752' 005767 173344            TST     S+10            ;IS THIS A DATA LATE ERROR?
 857  004756' 100012                   BPL     11$             ;NO
 858  004760' 005267 173436            INC     DLTCNT          ;ADD 1 TO DATA LATE COUNTER
 859  004764' 032767 000004 173024     BIT     #BIT2,@R1       ;TYPE ERROR AND COUNT IT?
 860  004772' 001103                   BNE     8$              ;NO
 861  004774' 104403 000000' 006730'   MSGN$,BEGIN,DLTERP      ;ASCII MESSAGE CALL WITH COMMON HEADER
 862  005002' 000450                   BR      5$              ;COM+
 863  005004' 032767 020000 173306 11$:BIT     #BIT13,S        ; MASSBUS CONTROL PARITY ERROR ?
 864  005012' 001035                   BNE     3$              ; YES
 865  005014' 032767 000400 173300     BIT     #BIT8,S+10      ; MASSBUS DATA PARITY ERROR ?
 866  005022' 001031                   BNE     4$              ; YES
 867  005024' 032767 040000 173326     BIT     #BIT14,S        ; TRANSFER ERROR ?
 868  005032' 001015                   BNE     2$              ; YES
 869  005034' 032767 040000 173262     BIT     #BIT14,S+12     ; ANY DRIVE ERRORS ?
 870  005042' 001030                   BNE     5$              ; YES
 871  005044' 005767 173260            TST     S+16            ; ANY ATTENTIONS ACTIVE ?
 872  005050' 001025                   BNE     5$              ; YES, CONTINUE
 873  005052' 005067 173230            CLR     ERRTYP          ;UNKNOWN ERROR
 874                                   ;*****************************************************************
 875  005056' 104405 000000' 001444'   ERDER$,BEGIN,TABLE      ; SPECIAL CONDITION SET BUT NO REASON FOUND
 876                                   ;*****************************************************************
 877  005064' 000446                   BR      8$              ; RETURN
 878  005066'                      2$:
 879  005066' 032767 100000 173266     BIT     #BIT15,S+42     ;IS A BAD SPOT ******* ?
 880  005074' 001045                   BNE     4$              ;BRANCH IF SO *****
 881  005076' 104403 000000' 006715'   MSGN$,BEGIN,TRERR       ;ASCII MESSAGE CALL WITH COMMON HEADER
 882  005104' 000407                   BR      5$              ; GO DUMP REGISTERS
 883  005106'                      3$:
 884  005106' 104403 000000' 006714'   MSGN$,BEGIN,MCERR       ;ASCII MESSAGE CALL WITH COMMON HEADER
 885  005114' 000403                   BR      5$              ; GO DUMP REGISTERS
 886  005116'                      4$:
 887  005116' 104403 000000' 006720'   MSGN$,BEGIN,MDPERR      ;ASCII MESSAGE CALL WITH COMMON HEADER
 888  005124' 005767 173200        5$: TST     S+16            ; ANY ATTENTIONS ACTIVE ?
 889  005130' 001004                   BNE     6$              ; YES, CONTINUE
 890  005132' 016777 173172 174376     MOV     S+16,@RMAS      ;KILL ATT BITS
 891  005140' 000205                   RTS     R5
 892  005142' 016700 174374        6$: MOV     RMDB,R0         ; SAVE ADDRESS OF DATA BUFFER
 893  005146' 105767 173150            TSTB    S+10            ; CAN DATA BUFFER BE READ ?
 894  005152' 100403                   BMI     7$              ; YES, CONTINUE
 895  005154' 012767 000430' 174360    MOV     #ZERO,RMDB      ; NO, LOAD ADDRESS OF ZERO
 896  005162' 012767 000001 172716 7$: MOV     #1,ERRTYP       ;DATA ERROR
 897                                   ;*****************************************************************
 898  005170' 104405 000000' 001444'   ERDER$,BEGIN,TABLE      ; DUMP RM11 AND RP REGISTERS
 899                                   ;*****************************************************************
 900  005176' 010067 174[illegible]    MOV     R0,RMDB         ; RESTORE DATA BUFFER ADDRESS
 901  005202' 004767 000[illegible] 8$: JSR     PC,NOTRDY       ;GO CLEAR OUT ERRORS
 902  005206' 000205                   RTS     R5              ; ERRORS REJECTED, RETURN
 903  005210' 005725                   TST     (R5)+           ;IF BAD SPOT DON'T REPORT
```

```
 904  005212  012767  177777  173222          MOV     #-1,BADSEC      ;SET BAD SECTOR DETECT FLAG
 905  005220  000205                          RTS     R5              ;EXIT*****
 906
 907
 908  005222  012767  077777  173204  NOTRDY: MOV     #77777,CLK      ; SET THE TIMER
 909  005230  016777  173166  174272  4$:     MOV     UNITNO,@RMCS2   ;MOVE DRIVE NUMBER TO RH11/70
 910  005236  012777  000011  174244          MOV     #11,@RMCS1      ;ISSUE DRIVE CLEAR COMMAND
 911  005244  032777  004000  174240          BIT     #BIT11,@RMCS1   ;DO I HAVE THE DRIVE DVA?
 912  005252  001406                          BEQ     6$              ;NO
 913  005254  010046                          MOV     R0,-(SP)
 914  005256  012600                          MOV     (SP)+,R0        ;JUST WASTE A LITTLE TIME
 915  005260  032777  004000  174230          BIT     #BIT11,@RMCS1           ;STILL GOT IT?
 916  005266  001023                          BNE     2$
 917  005270                          6$:
 918  005270  104407  000000  *               BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
 919  005274  104407  000000  *               BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
 920  005300  104407  000000  *               BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
 921  005304  104407  000000  *               BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
 922  005310  032777  010000  174210          BIT     #BIT12,@RMCS2   ;NED SET ?
 923  005316  001014                          BNE     5$              ;YES
 924  005320  005367  173110                  DEC     CLK             ;COUNT # OF TRIES
 925  005324  001341                          BNE     4$              ;NOT DONE YET
 926  005326  104403  000000  006734          MSG$S,BEGIN,NOT ;ASCII MESSAGE CALL WITH COMMON HEADER
 927  005334  000426                          BR      7$              ;COULD NOT GET DRIVE
 928  005336  004567  176416          2$:     JSR     R5,CLEAR        ;SET THE CONTROLLER AND DRIVE
 929  005342  004567  000112                  JSR     R5,READY        ; IS DRIVE READY ?
 930  005346  001443                          BR      1$              ; YES, CONTINUE
 931  005350  004767  000256          5$:     JSR     PC,ERSUB1       ; LOAD ERROR INFORMATION
 932  005354  005000                          CLR     R0              ;MOVE DRIVE REG INTO TABLE
 933  005356  016701  174136                  MOV     @RMCS1,R1
 934  005362  012160  000312          22$:    MOV     (R1)+,S(R0)     ;READ AND STORE ALL
 935  005366  005720                          TST     (R0)+           ;RH11/70 AND RM03
 936  005370  022700  000046                  CMP     #46,R0          ;REGISTERS
 937  005374  001372                          BNE     22$
 938  005376  012769  000006  172502          MOV     #6,ERRTYP       ;DRIVE NOT READY
 939                                          ;*****************************************************************
 940  005404  104405  000000  001444          ERDER$,BEGIN,TABLE      ; DRIVE  NOT READY
 941                                          ;*****************************************************************
 942  005412  012777  000013  174100  7$:     MOV     #13,@RMCS1      ;RELEASE DRIVE
 943  005420  004767  177224                  JSR     PC,DROP         ; NO, DROP THE DRIVE
 944  005424  104403  000000  006756          MSG$S,BEGIN,DRP ;ASCII MESSAGE CALL WITH COMMON HEADER
 945  005432  016706  172374                  MOV     SPOINT,R6
 946  005436  022767  000770  172742          CMP     #770,FLAG       ;IN PROCESS OF RETRIEVING BAD SPOT
 947  005444  001002                          BNE     .+6             ;BRANCH IF NOT
 948  005446  000167  174300                  JMP     RES2X           ;IF SO, BRANCH BACK
 949  005452  000167  174430                  JMP     LOOP2
 950  005456  000207                  1$:     RTS     PC              ;RETURN
 951                                          ;       ------------------------
 952
 953
 954                                  ;DVA=0,NED=0     NOT DETERMINE
 955                                  ;DVA=1,NED=0     DRIVE EXIST
 956                                  ;DVA=0,NED=1     DRIVE NOT EXIST
 957                                  ;DVA=1,NED=1     HARDWARE ERROR
 958
 959  005460  016777  172736  174042  READY:  MOV     UNITNO,@RMCS2   ; LOAD UNIT ADDRESS
```



```
 960  005466  012777  000011  174024          MOV     #11,@RMCS1      ;CLEAR DRIVE AND GRASP THE DRIVE
 961  005474  017700  174020                  MOV     @RMCS1,R0       ;R0= BASE ADDRESS OF RH CONTROLLER
 962  005500  032700  004000                  BIT     #BIT11,R0       ;DVA BIT = 1 ?
 963  005504  001426                          BEQ     1$              ;NO
 964  005506  017700  174                     MOV     @RMDS,R0        ; SAVE STATUS IN R0
 965  005512  105700                          TSTB    R0              ; DRIVE READY ?
 966  005514  100422                          BPL     1$              ; NO
 967  005516  032700  000040                  BIT     #BIT6,R0        ; VOLUME VALID ?
 968  005522  001417                          BEQ     1$              ; NO
 969  005524  032700  000400                  BIT     #BIT8,R0        ; DRIVE PRESENT ?
 970  005530  001414                          BEQ     1$              ; NO
 971  005532  032700  004000                  BIT     #BIT11,R0       ; WRITE LOCKED ?
 972  005536  001411                          BNE     1$              ; YES
 973  005540  032700  010000                  BIT     #BIT12,R0       ; MEDIUM ON LINE ?
 974  005544  001406                          BEQ     1$              ; NO
 975  005546  032700  040000                  BIT     #BIT14,R0       ; ANY ERRORS ?
 976  005552  001003                          BNE     1$              ; YES
 977  005554  005700                          TST     R0              ; ATTENTION SET ?
 978  005556  100411                          BMI     1$              ; YES
 979  005560  032777  004000  173732          BIT     #BIT11,@RMCS1   ;DVA SET?
 980  005566  001405                          BEQ     1$              ;BR IF NOT
 981  005570  032777  010000  173732          BIT     #BIT12,@RMCS2   ;NED BIT SET ?
 982  005576  001001                          BNE     1$              ;YES,ERROR
 983  005600  000205                          RTS     R5              ; RETURN READY
 984  005602  005725                  1$:     TST     (R5)+           ; SKIP INSTRUCTION FOLLOWING CALL
 985  005604  000205                          RTS     R5              ; RETURN AS NOT READY
 986                                          ;       ------------------------
 987
 988
 989  005606  014167  172274          ERSUB2: MOV     -(R1),ASB       ; LOAD THE DATA
 990  005612  010167  172264                  MOV     R1,SBADR        ; LOAD ADDRESS OF DATA WRITTEN
 991  005616  014267  172266                  MOV     -(R2),AWAS      ; LOAD THE DATA
 992  005622  010267  172256                  MOV     R2,WASADR       ; LOAD ADDRESS OF DATA READ
 993  005626  005741                          TST     (R1)+           ; RESET REG. 1
 994  005630  005742                          TST     (R2)+           ; RESET REG. 2
 995
 996  005632  016767  173662  172240  ERSUB1: MOV     RMCS1,CSRA      ; LOAD ADR OF CURRENT CSR
 997  005640  017767  173654  172234          MOV     @RMCS1,ACSR     ; LOAD CONTENTS OF CURRENT CSR
 998  005646  000207                          RTS     PC              ; RETURN
 999
1000
1001  005650  017700  173644          REZET:  MOV     @RMCS1,R0       ;DVA BIT OF RMCS1
1002  005654  032700  004000                  BIT     #BIT11,R0
1003  005660  001005                          BNE     3$
1004  005662  104407  000000  *               BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
1005  005666  104407  000000  *               BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
1006  005672  000766                          BR      REZET
1007  005674  012777  000040  173626  3$:     MOV     #BIT5,@RMCS2    ; ISSUE AN RM11 INIT
1008  005702  012767  077777  172554          MOV     #77777,CLK      ; SET THE TIMER
1009  005710  017700  173604          1$:     MOV     @RMCS1,R0               ; CONTROLLER READY ?
1010  005714  105700                          TSTB    R0
1011  005716  100417                          BMI     2$              ; YES, CONTINUE
1012  005720  104407  000000  *               BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
1013  005724  104407  000000  *               BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
1014  005730  005367  173560                  DEC     CLK             ; WAIT SOME MORE ?
1015  005734  001365                          BNE     1$              ; YES
```

```
                        ;7367 012767  000004  172142          MOV     #4,ERRTYP       ;CONTROLLER NOT READY
                        ;*****************************************************************
                        ;RDERS$,BEGIN,NULL                      ; CONTROLLER NOT READY
                        ;*****************************************************************
                        ;NDS,BEGIN                              ;
                        2$:     MOV     @RMAS,-(SP)             ;CLEAR ALL BITS
                                MOV     (SP)+,@RMAS             ;IN THE SUMMARY REGISTER
                                RTS     PC                      ; RETURN
                        ;       ----------------------------------

                        SETUP:  MOV     ADDR,R0                 ; GET DEVICE ADDRESS
                                MOV     R0,RMCS                 ; GENERATE REGISTER ADDRESSES
                                ADD     #2,R0
                                MOV     R0,RMWC
                                ADD     #2,R0
                                MOV     R0,RMBA
                                ADD     #2,R0
                                MOV     R0,RMDA
                                ADD     #2,R0
                                MOV     R0,RMCS2
                                ADD     #2,R0
                                MOV     R0,RMDS
                                ADD     #2,R0
                                MOV     R0,RMER1
                                ADD     #2,R0
                                MOV     R0,RMAS
                                ADD     #2,R0
                                MOV     R0,RMLA
                                ADD     #2,R0
                                MOV     R0,RMDB
                                ADD     #2,R0
                                MOV     R0,RMMR1
                                ADD     #2,R0
                                MOV     R0,RMDT
                                ADD     #2,R0
                                MOV     R0,RMSN
                                ADD     #2,R0
                                MOV     R0,RMOF
                                ADD     #2,R0
                                MOV     R0,RMDC
                                ADD     #2,R0
                                MOV     R0,RMHR
                                ADD     #2,R0
                                MOV     R0,RMMR2
                                ADD     #2,R0
                                MOV     R0,RMER2
                                ADD     #2,R0
                                MOV     R0,RMEC1
                                ADD     #2,R0
                                MOV     R0,RMEC2
                                RTS     PC
                        SETUP2: BIT     #1,FLAG         ;FIRST TIME THROUGH?
                                BEQ     1$              ;BR IF NOT
                                MOV     ADDR,R0         ;BASE ADDRESS OF RH11/RH70
                                ADD     #46,R0          ;INDEX VALUE
```



```
                                BIT     #ADDR22,RES1    ;11/70 SUPPORT?
                                BEQ     1$              ;NO
                                ADD     #2,R0
                                BIT     #BIT15,@R1      ;SPECIFY 32 REGISTER ON RH70 ?
                                BEQ     3$              ;BRANCH IF NOT
                                ADD     #24,R0          ;OTHERWISE ADJUST THE RMBAE ADDRESS
                        3$:     MOV     R0,RMBAE
                                ADD     #2,R0
                                MOV     R0,RMCS3
                        1$:     MOV     VECTOR,R0       ; GET VECTOR ADDRESS
                                MOV     #NTRUPT,(R0)+   ; SET POINTER JUST IN CASE
                                MOVB    #1,(R0)         ; SET PRICRITY
                        2$:     RTS     PC              ; RETURN

                        MES1:   .ASCIZ  "  TRANSFER  ERROR%"

                        MES2:   .ASCIZ  "  MASSBUS  PARITY  ERROR%"

                        MES3:   .ASCIZ  "  MASSBUS  DATA  PARITY  ERROR%"

                        MES4:   .ASCIZ  "  DRIVE  "

                        MES5:   .ASCIZ  "  DROPPED%"

                        MES6:   .ASCIZ  " RETRY EXCEEDED%"

                        MES10:  .ASCIZ  " DATA LATE ERROR%"

                        MES11:  .ASCIZ  " DRIVE NOT READY%"

                        MES12:  .ASCIZ  " COULD NOT GET DRIVE%"

                        MES14:  .ASCIZ  " OTHER PORT NOT UPDATING DRIVE%"

                        XMES15: .ASCIZ  "MORE 16 BAD BLOCKS DETECTED%"
```

```
1128    006664' 02 104  046102  041517
1129    006672' 051513  042040  042105
1130    006706' 041505  042524  022504
1131    006716'     00
1132            006710'                 .EVEN
1133    006710' 006334'                 TRERR:  MES1
1134    006712' 177777                          177777
1135    006714' 006357'                 MCPERR: MES2
1136    006716' 177777                          177777
1137    006720' 006411'                 MDPERR: MES3
1138    006722' 177777                          177777
1139    006724' 006476'                 EXCED:  MES6
1140    006726' 177777                          177777
1141    006730' 006517'                 DLTERR: MES10
1142    006732' 177777                          177777
1143
1144    006734' 006541'                 NOT:    MES11
1145    006736' 177777                          177777
1146    006740' 006563'                 TOUT:   MES12
1147    006742' 177777                          177777
1148
1149
1150    006744' 006451'                 PORTRG: MES4
1151    006746' 006777'                         NUMB
1152    006750' 006463'                         MES5
1153    006752' 006611'                         MES14
1154    006754' 177777                          177777
1155    006756' 006451'                 DRP:    MES4
1156    006760' 006777'                         NUMB
1157    006762' 006463'                         MES5
1158    006764' 177777                          177777
1159    006766' 006651'                 MES15:  XMES15
1160    006770' 177777                          177777
1161
1162    006772' 000001                  ADR1:   .BLKB   5
1163    006777'    00                   NUMB:   .BYTE   0
1164    007000'    00                           .BYTE   0
1165            007002'                         .EVEN
1166                                    ;       ------------------------
1167
1168            000001                          .END
```



```
ACSR    000102R          241#    997*
ADDR    000042R          207#   1027    1070
ADDR22= 001000           270#    694    1072
ADR1    006772R          838    1162#
ASB     000106R          245#    986*
ASTAT   000104R          243#
AWAS    000110R          246#    991*
BADSEC  000042R          318#    482     844*    904*
BADSPT  000252R          270#    388     410     413
BEGIN   000000R          204#    383     437     453     456     468     470     487     501     505     506     517     537
                         538     544     563     564     701     709     718     728     733     838     861     875     881
                         884     887     898     918     919     920     921     926     940     944    1004    1005    1012
                        1013    1018    1020
BITTAB  004542R          712     788     801#
BIT0    = 000001         270#
BIT1    = 000002         270#
BIT10   = 002000         270#
BIT11   = 004000         270#    911     915     962     971     979    1002
BIT12   = 010000         270#    690     922     973     981
BIT13   = 020000         270#    863
BIT14   = 040000         270#    689     867     869     975
BIT15   = 100000         270#    879    1075
BIT2    = 000004         270#    859
BIT3    = 000010         270#
BIT4    = 000020         270#
BIT5    = 000040         270#   1007
BIT6    = 000100         270#    987
BIT7    = 000200         270#
BIT8    = 000400         270#    865     969
BIT9    = 001000         270#
BLK     000037R          300#    342     368     375*    415*    418*    423*    427*    430     442*    494     551*    752
                         766    807*    808     810*    815
BLKADR  004252R          480     552     747#
BREAKS= 104407           270#    506     506     537     538     563     564     918     919     920     921    1004    1005
                        1012    1013
BR1     000012R          269#   1083
BR2     000015R          270#
BTODS = 104421           270#
CDATAS= 104412           270#    487     517
CDBRCT  000144R          260#
CDWDCT  000146R          261#
CLEAR   003660R          679#    928
CLK     000434R          315#    908*    924*   1008*   1014*
CLRCOM  004640R          823#    825
CLRRP   004630R          462     556     821#
CNT     000416R          308#    367     384#    449     454*
CNT1    000420R          309#    447     469*
CONFIG  000156R          229#
CSRA    000100R          239#    996*
CYA     002402R          477     479#
CYB     002440R          523     525#    547     549
CYCLE   002360R          472     474#
CYCLER  002620R          475     521#
CYL     000422R          302#    397*    586     602     618     634     650     749*    755*    769*
DATCKS= 104411           270#
DATERS= 104404           270#
```

```
DLTCNT  000422R          310#    858*
DLTERR  000736R          861    1141#
DRIVE   000412R          308#
DROP    004650R          500     543     732     828#    943
DRP     000756R          733     944    1155#
DSKADR  000440R          301#    398*    399#    585     601     617     633     649     748*    763*    764*    777*    778*
DVICE   000410R          305#    373*    466     788     834*
DVID1   000014R          211#    373
EA22    000374R          298#    703     704*    705*    706
ENDITS= 104413           270#    470
ENDS    = 104410         270#    468    1020
ERRORS  000470R          721     844#
ERRTYP  000106R          244#    873*    896*    938*    1016*
ERSUB1  000532R          865     931     996#
ERSUB2  000560R          899#
EXCED   000724R          728    1139#
EXITS = 104400           270#    705
FERADR  000432R          314#    366     580*    596*    612*    628*    644*    660*    676*
FLAG    000406R          304#    376*    396*    443*    465*    479*    493     526     557*    558     736     946     1068
FREE    000150R          265#
FUNC    000424R          311#    579*    595*    611*    627*    643*    659*    675*    707*    708
GETPAS= 104415           270#    383     453
GO      003744R          594     610     626     642     658     674     693#
GWBUFS= 104414           270#    456
HRDCNT  000044R          224#
HRDERS= 104405           270#    875     898     940     1018
HRDPAS  000050R          226#
ICONT   000036R          221#
ICOUNT  000040R          222#
IDNUM   001122R          251#
IMODX.= 000000           263#    457
INIT    000030R          218#
INTP    000120R          250#    372*
LOOP1   002270R          386     455#    472
LOOP2   002306R          458#    463     735     741     949
MAP22S= 104416           270#    701
MCPERR  006714R          884    1135#
MDPERR  006720R          887    1137#
MES1    006334R         1086#   1133
MES10   006517R         1108#   1141
MES11   006541R         1112#   1144
MES12   006563R         1116#   1146
MES14   006611R         1120#   1153
MES15   006766R          437    1159#
MES2    006357R         1090#   1135
MES3    006411R         1096#   1137
MES4    006451R         1101#   1150    1155
MES5    006463R         1103#   1152    1157
MES6    006477R         1105#   1136
MIXDV   003044R          317#    403     750     782*    799*    811
MONNAM  000000R          265#
MODSP   002252R          219     268#
MSGNS = 104403           270#    437     501     544     728     733     861     881     884     887     926     944
MSGSS = 104402           270#
MSGS  = 104401           270#
MOT     006734R          476    1144#
```



```
NOTRDY  005222R          402     478     511     524     548     794     901     908#
NTRUPT  004072R          710#   1082
NULL  = 000000           270#   1018
NUMR    006777R         1151    1156   1163#
PDN   = 000000           206     212     213     214     215     232     233     234     235     236     237     238     239
                         241     243     245     246     248     249     250     253     254     256     257     259     260
                         262     270#
                         277#    836
TOAS  = 104420           277#
ASCNT   000034R          224#
A18     000366R          296#    696*    701
A22     000372R          298#    702
ICKBK   004552R          457     739     807#    816
ICKDP   004434R          379     393     459     781#
IRQS  = 000004           270#    718
OPSP  = 005726           270#
OPSP2 = 022626           270#
ORTHG   006744R          541     544    1150#
RTV   = 000000           270#
RTV0  = 000000           270#    270#
RTV1  = 000040           270#
RTV2  = 000100           270#
RTV3  = 000140           270#
RTV4  = 000200           270#
RTV5  = 000240           270#    270#
RTV6  = 000300           270#
RTV7  = 000340           270#
S     = 177776           270#
SW    = 177776           270#
USH   = 005746           270#
USH2  = 004646           270#
ANDS  = 104417           270#
ANNUM   000054R          428#
BUF     002444R          252     319#    406     493*    494*    514     528     530     532     551     558*    821
BUFFA   001130R          254#    626     636     652     668
BUFPA   001126R          253#    487     517     616     632     648     664
BUFSZ   001132R          255#    613     629     645     661     822
BUFVA   001124R          252#    383     453
DDQ     003642R          523     525     675#    676
EAD     003262R          486     516     553     611#    612
EADV    005460R          466     476     509     522     546     792     929     959#
ESTPT   002236R          247     447#
ESI     000056R          230#    644    1072
ES2     000060R          231#
ES2X    001752R          355#    441     738     948
EZET    000650R          342    1001#   1006
MAS     001536R          351#    687     680*    713     715*    890*    1021    1022*   1042*
MBA     001524R          346#    584*    600*    616*    632*    648*    664*    696     702*    1032*
MBAE    001570R          364#    703*   1078*
MCSI    001520R          344#    497*    532*    560*    680*    683*    684     689*    708*    845     850     910*    911
                         915     933     942*    960*    961     979     996     997     1001    1009    1028*
MCS2    001530R          345#    679*    693*    909*    922     959*    981     1007*   1036*
MCS3    001572R          365#   1040*
MDA     001526R          347#    585*    601*    617*    633*    649*    665*    1034*
MDB     001542R          353#    892     895*    900*   1046*
MDC     001554R          358#    586*    602*    618*    634*    650*    666*    1056*
MDS     001532R          348#    864    1038*
```

```
RMDT    001546R          355#    795     797    1050*
RMEC1   001564R          363#   1064*
RMEC2   001566R          364#   1066*
RMER1   001534R          353#   1040*
RMER2   001562R          361#   1062*
RMHR    001556R          359#   1058*
RMLA    001540R          352#   1044*
RMMR1   001544R          354#   1046*
RMMR2   001560R          360#   1060*
RMOP    001552R          357#    690*  1054*
RMSN    001550R          356#   1052*
RMWC    001522R          345#    583*   599*   615*   631*   647*   663*  1030*
RSTRT   000112R          247#
RSTRT1  002256R          445     448    450    452#
S       000312R          286#    322    323    324    325    326    327    328    329    330    331    332    333
                         334     335    336    337    338    339    340    341    851*   856    863    865    867
                         865     871    879    888    890    893    934*
SBADR   000102R          240*    990*
SETUP   005770R          378    1027#
SETUP2  006232R          381    1068#
SOFCNT  000042R          223*
SOPERS= 10440R           670*
SOFPAS  000046R          225#
SPOINT  000032R          219#    374     734     945
SPSIZ = 000040             1#    263
SR1     000016R          212#    385     474     730     850    1075
SR2     000020R          213#
SR3     000022R          214#
SR4     000024R          215#
START   001604R          218     370#    451
STAT    000026R          217#
SVR0    000062R          232#
SVR1    000064R          233#
SVR2    000066R          234#
SVR3    000070R          235#
SVR4    000072R          236#
SVR5    000074R          237#
SVR6    000076R          238#
SYSCNT  000052R          227#
TABLE   011444R          321#    875     898     940
TIMER   000426R          312#    496*    498*    521*    541*
TOUT    006740R         1144#
TRACK   000404R          303#    747*    760*    764     774*    778
TRERR   006710R          891    1133#
TRPDFD= 000022           278#
TRY     000436R          316#    395*    461*    725*    726
UNITNO  000414R          307#    377*    387*    455*    679     693     711     740*    783*    784     787     829     838
                         909     959
VECTOR  000010R          208#   1081
WASADR  000104R          242#    992*
WBUFEA  000136R          257#    580     604
WBUFPA  000134R          256#    584     600
WBUFRQ  000140R          258#
WBUFSZ  000142R          259#    581     597
WDFR    000116R          249#    371*
WDTO    000114R          248#    370*
```



```
WRITCR  003452R          555     643#    644
WRITCK  003166R          485     595#    596
WRITE   003072R          481     579#    580
WRITER  003356R          554     627#    628
WRTOO   003546R          495     559     659#    660
WRTOOP  003620R          661*    677
XBLK    001660R          368#
XFERAD  001574R          366#
XFERCT  001576R          367#
XFLAG   000054R          206#
XMEM    003700R          597#    593*    609*    625*    641*    657*    673*    697*    698*    699*    700*    706*    707
XMES15  006651R         1126#   1159
ZERO    000430R          313#    495
.       = 007002R        319#    401     483     488     518     737     947    1132#   1162#   1165#

. ABS.  000000      000
        007002      001

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

XRMBC0/XRMBC0/SOL/CRF:SYM=DDXCOM,XRMBC0
RUN-TIME:  2  3  .5 SECONDS
RUN-TIME RATIO: 27/5=4.7
CORE USED:  7K  (13 PAGES)
```