RM01

FORMATTER PROGRAM

MD-11-DZRMA-A

EP-DZRMA-A-DL-A

FICHE 1 OF 1

MAR 1977

digital

MADE IN USA

B01

E:F:DZI*4DSE3	00010000	770224	PDP10 411	1HDR1DZRMAASE@	00010000	770224

```
                        .REM    %
```

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

PRODUCT CODE: MD-11-DZRMA-A

PRODUCT NAME: RM01 FORMATTER PROGRAM

DATE CREATED: JANUARY 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. HESS/C. CHEN

CONTENTS

```
                        1.      ABSTRACT
                                --------

                        THE RM01 FORMATTER PROGRAM PROVIDES THE FACILITIES  TO FORMAT OR
                        TO VERIFY THE HEADER AND DATA FIELDS OF EACH DATA BLOCK ON THE DISC PACK.
                        EACH HEADER FIELD SPECIFIES THE ADDRESS OF ITS ASSOCIATED DATA
                        BLOCK ON THE DISK PACK.


                        IN THE FORMAT OPERATION,THE PROGRAM WRITES THE HEADER OF EACH
                        DATA BLOCK WITH A CYLINDER NUMBER,TRACK NUMBER AND SECTOR NUMBER;
                        WRITES THE DATA FIELD WITH SELECTED DATA PATTERN.



                        IN THE VERIFICATION OPERATION,THE PROGRAM ASSUMES THAT THE DISK PACK
                        HAS BEEN FORMATTED. THEN,THE PROGRAM CHECKS THE HEADER AND DATA OF EACH
                        DATA BLOCK.  IF ANY ERROR ARISES,THE PROGRAM WRITES THE CORRECT
                        ADDRESS INFORMATION ON THE HEADER.


                        2.      REQUIREMENTS
                                ------------

                        2.1     EQUIPMENT

                                PDP-11 PROCESSOR
                                8K MEMORY
                                TELETYPE
                                PROGRAM LOAD DEVICE
                                KW11-L OR KW11-P CLOCK
                                RH11 OR RH70 WITH 1 - 8 RM01 DISK DRIVES
                                DISK PACK(S)

                        2.2     PRELIMINARY PROGRAMS

                                THERE ARE NO PREQUISITE PROGRAMS PROVIDING THE RM01 SUBSYSTEM IS
                                KNOWN TO BE OPERATIONAL.  IF THE STATE OF THE RM01 SUBSYSTEM IS UNKNOWN,
                                THE FOLLOWING PROGRAMS SHOULD BE RUN PRIO TO USING THE FORMATTER PROGRAM
                                        RM01    DISKLESS DIAGNOSTIC
                                        RM01    FUNCTIONAL TEST


                        3.      LOADING PROCEDURES
                                ------------------

                        THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE
                        LOADER OR IT MAY BE LOADED FROM 'XXDP' MEDIA USING THE APPROPRIATE
                        LOADER.  THE PROGRAM MAY NOT BE INCLUDED IN AN 'XXDP' CHAIN.



                        4.      STARTING PROCEDURES
                                -------------------
```

4.1 STARTING ADDRESSES

THE PROGRAM IS STARTED FROM LOCATION 200(8) IF THE ADDRESS OF THE RH11 OR RH70 WILL NOT BE CHANGED FROM ITS DEFAULT VALUE OF 176700.

THE PROGRAM IS STARTED FROM LOCATION 204(8) IF THE ADDRESS OF THE RH11 OR RH70 IS TO BE CHANGED FROM THE PRELOADED VALUE. NOTE THAT STARTING ADDRESS 204 IS VALID ONLY ONCE AFTER THE PROGRAM IS LOADED. (SEE SECTION 4.3)

4.2 OPERATION ACTION

1. LOAD THE PROGRAM INTO MEMORY (SEE SECTION 3).

2. LOAD THE STARTING ADDRESS - 200(8) OR 204(8).

3. SET THE SWITCHES AS REQUIRED AND PRESS 'START'.

   IF THIS IS THE PROGRAM'S FIRST START, THE STATUS OF THE DRIVES ON THE SELECTED MASSBUS SUBSYSTEM WILL BE TYPED OUT. THIS TYPEOUT MAY BE INHIBITED ON SUBSEQUENT STARTS BY SETTING SW(02).

4. THE PROGRAM WILL THEN TYPE THE FOLLOWING MESSAGE:

   'PROGRAM MODE (C OR F):'

   ENTER THE APPROPRIATE CODE: 'C' FOR 'CHECK' OPERATION OR 'F' FOR 'FORMAT & VERIFY'. IF A 'CARRIAGE RETURN' IS ENTERED IN RESPONSE TO THE REQUEST, THE PROGRAM WILL ASSUME 'FORMAT & VERIFY'.

5. THE PROGRAM WILL THEN ASK FOR THE FORMATTING MODE:

   'OPERATE IN 32 SECTOR (16 BIT) MODE (Y OR N)'

   ENTER THE APPROPRIATE CHARACTER: 'Y' FOR 16 BIT MODE OR 'N' FOR 18 BIT MODE. IF A 'CARRIAGE RETURN' IS ENTERED IN RESPONSE TO THIS REQUEST, THE PROGRAM WILL ASSUME 16 BIT MODE.

6. THE PROGRAM WILL THEN ASK FOR A DRIVE:

   'DRIVE: '

   ENTER THE ADDRESS OF THE DRIVE TO BE FORMATTED. A 'PERIOD' OR 'CARRIAGE RETURN' ENTRY WILL SELECT DRIVE 0. IF THE DRIVE SELECTED IS NOT AVAILABLE, THE PROGRAM WILL TYPE AN ERROR MESSAGE AND RETURN TO THE DRIVE ADDRESS REQUEST.

7. AFTER THE OPERATOR HAS SELECTED A DRIVE, THE PROGRAM WILL ASK FOR ADDRESS LIMITS FOR THE SELECTED DRIVE:

   'ENTER ADDRESS LIMITS: '

   THE PREVIOUSLY SELECTED OR DEFAULT VALUES FOR BEGINNING CYLINDER

AND TRACK AND FOR ENDING CYLINDER AND TRACK WILL BE TYPED OUT.
IF A 'CARRIAGE RETURN' IS TYPED AS A RESPONSE, THE PRESENT VALUE
WILL BE USED; IF A 'PERIOD' IS TYPED, THE PROGRAM WILL BYPASS
THE REMAINING ENTRIES AND WILL USE THEIR PRESENT VALUES.
NOTE THAT THE CYLINDER AND TRACK VALUES ARE D E C I M A L NUMBERS.
THE ADDRESS SPECIFIED BY THE BEGINNING TRACK MUST BE
LESS THAN THE ENDING TRACK ADDRESS IF THESE TWO
ADDRESSES ARE ON THE SAME CYLINDER. ON THE OTHER HAND,
THE ENDING CYLINDER ADDRESS MAY BE LESS THAN THE STARTING
CYLINDER ADDRESS. IN THIS CASE ,THE PROGRAM WILL FORMAT
OR VERIFY THE DISK PACK FROM THE STARTING CYLINDER TO
CYLINDER 822 AND THEN FROM CYLINDER 0 TO THE ENDING CYLINDER.

8. THE PROGRAM WILL THEN ASK FOR THE DATA PATTERN:

```
'SELECT DATA PATTERN
  (0) ZERO'S AND ONE'S
  (1) AB AND CD
  (2) WORST CASE:'
```

ENTER THE CODE FOR THE REQUIRED PATTERN. 'CARRIAGE RETURN' OR
'PERIOD' ENTRIES WILL CAUSE THE PROGRAM TO USE THE 'WORST CASE'
PATTERN.

THE 'WORST CASE' PATTERN IS THE FOLLOWING OCTAL SEQUENCE REPEATED
THROUGH THE DATA AREA OF THE SECTOR:

```
165555
133333
```

ALTERNATE CYLINDERS ARE RECORDED WITH COMPLEMENT OF THE
SELECTED DATA PATTERN.

9. THE PROGRAM WILL THEN TYPE:

'STARTING FORMAT ON DRIVE N'

OR

'STARTING CHECK ON DRIVE N'

10. THE OPERATOR CAN DETERMINE WHERE THE DRIVE IS DURING THE FORMAT
OR CHECK OPERATION BY TYPING A 'CONTROL O'. THE PROGRAM WILL
TYPE THE FOLLOWING MESSAGE:

'PRESENT ADDRESS IS: CXXX TXX'

IF A 'CONTROL C' IS TYPED WHILE THE PROGRAM IS TYPING THE CURRENT
ADDRESS, THE PROGRAM WILL ABORT THE FORMAT (OR CHECK) OPERATION
AND RETURN TO THE 'DRIVE' REQUEST.

11. IF THE OPERATOR DOES NOT SELECT A DIFFERENT DRIVE WHEN THE FORMAT
OR CHECK OPERATION HAS COMPLETED, THE PROGRAM WILL NOT ALTER THE
ADDRESS LIMITS SPECIFIED AT THE START OF THE PREVIOUS OPERATION.
IF THE FORMAT OR CHECK OPERATION IS TERMINATED BY A 'CONTROL C' OR

IF A DIFFERENT DRIVE IS SELECTED, THE PROGRAM WILL RESET THE ADDRESS LIMITS TO THE VALUES APPROPRIATE FOR THE DRIVE TYPE.

12. TO CHANGE EITHER THE ADDRESSING MODE OR THE OPERATION MODE, THE PROGRAM MUST BE STARTED FROM LOCATION 200(8) OR 204(8) AGAIN.

4.3 RH11 - RH70 UNIBUS ADDRESS

THE PROGRAM ASSUMES THAT THE RH11 OR RH70 ADDRESSES START AT 176700 AND THAT THE VECTOR ADDRESS IS 254. THESE ADDRESSES MAY BE CHANGED WHEN THE PROGRAM IS STARTED FROM LOCATION 204(8). IF THE RH11 - RH70 IS NOT AT THE DEFAULT ADDRESS, THE PROGRAM MUST BE STARTED FROM 204(8) INITIALLY AS THE PROGRAM GOES THROUGH THE ADDRESS CHANGE ROUTINE AT INITIAL START ONLY. ENTER THE RH11 RH7C ADDRESS IN RESPONSE TO THE REQUEST FROM THE PROGRAM.

4.4 OTHER UNIBUS ADDRESSES

LOC	MG	CONTENTS	FUNCTION
1144	$TKS	177560	TTY KEYBOARD STATUS REGISTER
1146	$TKB	177562	TTY KEYBOARD BUFFER REGISTER
1150	$TPS	177564	TTY PRINTER STATUS REGISTER
1152	$TPB	177566	TTY PRINTER BUFFER REGISTER
1176	$LKCSR	172540	KW11-P CONTROL REGISTER
1200	$LKCSB	172542	KW11-P COUNTER REGISTER
1202	$LPVEC	104	KW11-P VECTOR ADDRESS
1206	$LKS	177546	KW11-L CONTROL REGISTER
1210	$LKV	100	:ADDRESS OF KW11-L VECTOR

5. SWITCH REGISTER SETTINGS

SW<15>=1...HALT ON ERROR
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<10>=1...BELL ON ERROR
SW<09>=1...LOOP ON ERROR
SW<02>=1...DON'T DISPLAY SYSTEM STATUS AFTER INITIAL START
SW<01>=1...LOOP ON THE CURRENT TRACK
SW<00>=1...LOOP THE PROGRAM ON THE SELECTED DRIVE

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34, THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RM01 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

6. ERROR MESSAGES

1. 'RH11 INTERRUPT OCCURRED (RMAS=0) - AN INTERRUPT OCCURRED, BUT NOTHING ON THE MASSBUS IS INDICATING AN ATTENTION.

2. 'UNEXPECTED ATTENTION OCCURRED' - THE INDICATED DRIVE INTERRUPTED, BUT NO INTERRUPT WAS EXPECTED FROM THE INDICATED DRIVE.

3. 'MASSBUS PARITY ERROR (MCPE=1)' - A CONTROL BUS PARITY ERROR WAS DETECTED BY THE RH11 WHEN THE INDICATED REGISTER WAS READ.

4. 'MASSBUS PARITY ERROR (PAR=1)' - A CONTROL BUS PARITY ERROR OCCURRED WHEN THE INDICATED REGISTER WAS WRITTEN.

5. 'ADDRESS PLUG CHANGE BIT SET' - THE PROGRAM FOUND THE 'OPE' BIT SET FOR THE INDICATED DRIVE.

6. 'RH11 DIDN'T RESPOND TO ADDRESSING' - THE PROGRAM ACCESSED THE RH11 AT THE INDICATED ADDRESS AND RECEIVED NO RESPONSE.

7. 'DRIVE OFFLINE' - THE INDICATED DRIVE HAS GONE OFFLINE

8. 'PERSISTENT DRIVE UNSAFE ERROR' - THE INDICATED DRIVE HAS BECOME UNSAFE AND THE CONDITION CANNOT BE CLEARED BY ISSUING 'DRIVE CLEAR' INSTRUCTIONS.

9. 'UNCORRECTABLE MASSBUS PARITY ERROR' - THE PROGRAM ATTEMPTED TO PERFORM AN OPERATION AND DETECTED 3 SUCESSIVE MASSBUS PARITY ERRORS OR THE PROGRAM ATTEMPTED TO CLEAR A 'PAR' ERROR IN THE DRIVE AND A PARITY ERROR OCCURRED.

10. 'SOFTWARE TIMEOUT' - THE OPERATION FAILED TO COMPLETE WITHIN 1 SECOND.

11. 'DRIVE UNSAFE ERROR' - A NON-PERSISTENT UNSAFE ERROR OCCURRED DURING THE OPERATION.

12. 'CONTROLLER/DRIVE ERROR DURING WRITE' - THE INDICATED NON-DATA ERROR WAS DETECTED DURING A FORMAT OPERATION.

13. 'CONTROLLER/DRIVE ERROR DURING WRITE CHECK' - A NON-DATA ERROR OCCURRED DURING THE WRITE CHECK.

14. 'DATA ERROR DURING WRITE CHECK' - A DATA RELATED ERROR OCCURRED DURING A WRITE CHECK OPERATION. A DATA ERROR IS CONSIDERED TO BE ONE OF THE FOLLOWING ERRORS: 'DCK', 'OPI', 'DTE', 'HCRC', 'HCE', OR 'FER'.

15. 'RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE' - THE INDICATED SECTOR FAILED DURING RETRY FOLLOWING A DATA ERROR.

16. 'CONTROLLER/DRIVE ERROR VERIFYING HEADERS' - AN ERROR OCCURRED DURING THE VERIFICATION PASS AFTER FORMATTING.

17. ''HCE' ERROR VERIFYING HEADERS' - A HEADER ERROR OCCURRED DURING THE VERIFICATION PASS AFTER FORMATTING.

18. 'CYLINDER FIELD IN HEADER IS NOT CORRECT' - THE CYLINDER FIELD FROM A HEADER READ DURING THE VERIFICATION PASS IS NOT CORRECT.

19. 'WRITE CHECK ERROR' - THE RH11 REPORTED A WRITE CHECK ERROR AND NO DRIVE ERRORS WERE SET.

7. MISCELLANEOUS

7.1 FORMAT TIME

IT TAKES APPROXIMATELY 16 MINUTES TO FORMAT AN ENTIRE RM01 PACK. THE 'CHECK' MODE TIME FOR AN ENTIRE RM01 PACK IS 8 MINUTES.

7.2 HALTING THE PROGRAM

THE OPERATOR SHOULD NOT HALT THE PROGRAM DURING A FORMAT OPERATION. HALTING THE PROGRAM MAY LEAVE A SECTOR INCORRECTLY FORMATTED. TO TERMINATE THE FORMAT, TYPE A 'CONTROL O' AND WHILE THE PROGRAM IS TYPING THE ADDRESS, TYPE ANOTHER 'CONTROL C'; THIS SEQUENCE RETURNS THE PROGRAM TO THE DRIVE ADDRESS ENTRY ROUTINE.

7.3 SURFACE VERIFICATION

THE FORMATTER PROGRAM IS NOT INTENDED TO BE USED TO PERFORM DISK PACK VERIFICATION. IF THE PROGRAM REPORTS A SECTOR AS BEING 'NOT ACCEPTABLE', THIS MAY IN FACT INDICATE A BAD SURFACE; HOWEVER, SECTORS WHICH 'PASSED' MAY OR MAY NOT BE GOOD.

8. PROGRAM DESCRIPTION

8.1 FORMAT OPERATION

THE PROGRAM FORMATS THE PACK ONE TRACK AT A TIME BETWEEN THE LIMITS SPECIFIED BY THE OPERATOR.

THE FORMAT OPERATION CONSISTS OF A WRITE HEADER AND DATA COMMAND FOR THE ENTIRE TRACK FOLLOWED BY A WRITE CHECK HEADER AND DATA COMMAND. IF AN ERROR OCCURS DURING THE WRITE, THE PROGRAM WILL RETRY THE WRITE BEFORE CONTINUING. IF A DATA RELATED ERROR IS DETECTED DURING THE WRITE CHECK (A DATA ERROR IS DEFINED AS ONE OF THE FOLLOWING ERRORS: 'DCK', 'OPI', 'DTE', 'HCRC', 'HCE', OR 'FER'), THE SECTOR IN ERROR WILL BE REWRITTEN. THE PROGRAM WILL CHECK THE SECTOR TWICE; IF A DATA ERROR IS DETECTED IN THE SECTOR DURING EITHER OF THE WRITE CHECKS, THE PROGRAM WILL DECLARE THE SECTOR AS BEING 'NONRECOVERABLE'. FOLLOWING THIS SEQUENCE, THE REMAINDER OF THE TRACK IS CHECKED. IF DATA ERRORS ARE ENCOUNTERED IN ANY OF THE REMAINING SECTORS, EACH SECTOR WILL BE HANDLED AS DESCRIBED ABOVE.

IF A NON-DATA RELATED ERROR OCCURS DURING THE WRITE CHECK, THE PROGRAM WILL RETRY THE COMMAND FOR THE ENTIRE TRACK BEFORE PROCEEDING TO THE NEXT TRACK.

IT SHOULD BE NOTED THAT THE FORMATTER PROGRAM FORMATS THE PACK ONLY AND IS NOT DIRECTLY VERIFYING THE SURFACE OF THE PACK. ERRORS THAT ARE ENCOUNTERED MAY BE THE RESULT OF BAD AREAS ON THE PACK BUT, AS THE PROGRAM IS NOT DESIGNED TO PERFORM AN EXHAUSTIVE CHECK OF THE DISK PACK SURFACE, THE PROGRAM CANNOT MAKE THIS CONCLUSION. IN GENERAL, HOWEVER, THE OPERATOR CAN ASSUME THAT SECTORS WHICH THE PROGRAM CALLS 'UNACCEPTABLE' INDICATE BAD AREAS OF THE PACK; UNFORTUNATELY, SECTORS WHICH 'PASS' CANNOT BE ASSUMED TO BE GOOD.

DURING THE FORMAT OPERATION, THE PROGRAM FILLS THE DATA FIELD WITH THE PATTERN SELECTED BY THE OPERATOR.

8.2 CHECK OPERATION

THE CHECK OPERATION IS IDENTICAL TO THE WRITE CHECK PORTION OF THE FORMAT OPERATION OPERATION DESCRIBED IN SECTION 8.1 EXCEPT THAT THE PROGRAM WILL NOT RE-WRITE ERROR SECTORS.

8.3 POSITIONER VERIFICATION

AFTER THE PROGRAM COMPLETES THE FORMAT OPERATION, THE POSITIONER IS RETURNED TO THE STARTING CYLINDER AND THE HEADER FROM SECTOR 0 ON THE STARTING TRACK IS READ. THE CYLINDER ADDRESS FIELD FROM THE HEADER IS COMPARED TO THE REQUESTED CYLINDER; IF THE CYLINDER ADDRESSES DO NOT COMPARE, AN ERROR MESSAGE IS TYPED. THE PROGRAM CHECKS THE CYLINDER ADDRESS FIELD FROM THE HEADER OF SECTOR 0 ON THE BEGINNING TRACK OF EACH CYLINDER FORMATTED. THIS CHECK IS PERFORMED TO CONFIRM THAT THE DISK'S POSITIONER ADVANCED PROPERLY DURING THE FORMAT OPERATION.

477

```
479
479                                     %
480
481
```

```
                                .TITLE  MD-11-DZRMA, RM01 FORMATTTER PROGRAM
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY C. HESS/C. CHEN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                                ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         2            DON'T DISPLAY SYSTEM STATUS AFTER INITIAL START
                                ;*         1            LOOP ON THE CURRENT TRACK
                                ;*         0            LOOP THE PROGRAM ON THE SELECTED DRIVE

                                .SBTTL  TRAP CATCHER

                000000                  .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILL GAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                  .=174
000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
                000200                  $SVPC=.                 ;SAVE PC
                000046                  .=46
000046  005432                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SECP
                000052                  .=52
000052  020000                          .WORD   20000           ;;2)SET LOC.52 TO 20000
                000200                  .=$SVPC                 ;; RESTORE PC

                                .SBTTL  STARTING ADDRESS = 200
                000200                  .=200
000200  000137  002150                  JMP     BEGIN1          ;NORMAL STARTING ADDRESS

                                .SBTTL  STARTING ADDRESS TO CHANGE THE RH11 ADDRESS =*204
000204  000137  002140                  JMP     BEGIN           ;CHANGE THE RH11 ADDRESS

                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
                001100          STACK=  1100
```

```
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
000002                          SW01=   2
000001                          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
```

```
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
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
                                .EQUIV  BIT09,BIT9
                                .EQUIV  BIT08,BIT8
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;"T" BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
000014                          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                          PWRVEC= 24              ;;POWER FAIL
000030                          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                          TRAPVEC=34              ;;"TRAP" TRAP
000060                          TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                          TPVEC=  64              ;;TTY PRINTER VECTOR
000240                          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

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

                                .SBTTL  RH11 REGISTERS

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

```
                                ;CONTROL AND STATUS REGISTER 1 (RMCS1)

000100                  IE=     100             ;INTERRUPT ENABLE (BIT #6)
000200                  RDY=    200             ;READY (BIT #7)
000400                  A16=    400             ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
001000                  A17=    1000            ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
002000                  PSEL=   2000            ;PORT SELECT (BIT #10)
020000                  MCPE=   20000           ;MASSBUSS PARITY ERROR (BIT #13)
040000                  TRE=    40000           ;TRANSFER ERROR (BIT #14)
                        ;SC=    100000          ;SPECIAL CONDITION (BIT #15)

                        ;WORD COUNT REGISTER (RMWC)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;BUS ADDRESS REGISTER (RMBA)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;CONTROL AND STATUS REGISTER 2 (RMCS2)

000001                  US1=    1               ;UNIT SELECT (BIT #0)
000002                  US2=    2               ;UNIT SELECT (BIT #1)
000004                  US4=    4               ;UNIT SELECT (BIT #2)
000010                  BAI=    10              ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
000020                  PAT=    20              ;MASSBUS PARITY TEST (BIT #4)
000040                  CLR=    40              ;CLEAR (BIT #5)
000100                  IR=     100             ;INPUT READY (BIT #6)
000200                  OR=     200             ;OUTPUT READY (BIT #7)
000400                  MPE=    400             ;MASS BUS PARITY ERROR (BIT #8)
001000                  MXF=    1000            ;MISSED TRANSFER ERROR (BIT #9)
002000                  PGE=    2000            ;PROGRAM ERROR (BIT #10)
004000                  NEM=    4000            ;NON EXISTENT MEMORY (BIT #11)
010000                  NED=    10000           ;NON EXISTENT DRIVE (BIT #12)
020000                  UPE=    20000           ;UNIBUS PARITY ERROR (BIT #13)
040000                  WCE=    40000           ;WRITE CHECK ERROR (BIT #14)
100000                  DLT=    100000          ;DATA LATE (BIT #15)

                        ;DATA BUFFER REGISTER (RMDB)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)


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

                        .SBTTL  RM01 REGISTERS

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

                        ;CONTROL AND STATUS 1 REGISTER. (#00)

000001                  GO=     1               ;GO BIT (BIT #0)
000002                  F1=     2               ;FUNCTION CODE BIT #1
000004                  F2=     4               ;FUNCTION CODE BIT #2
000010                  F3=     10              ;FUNCTION CODE BIT #3
000020                  F4=     20              ;FUNCTION CODE BIT #4
000040                  F5=     40              ;FUNCTION CODE BIT #5
004000                  DVA=    4000            ;DEVICE AVAILABLE (BIT #11)
```

```
                        ;DRIVE STATUS REGISTER (RMDS1) (#01)

000100          VV=     100             ;VOLUME VALID (BIT #6)
000200          DRY=    200             ;DRIVE READY (BIT #7)
000400          DPR=    400             ;DRIVE PRESENT (BIT #8)
001000          PGM=    1000            ;PROGRAMABLE (BIT #9)
002000          LST=    2000            ;LAST SECTOR TRANSFERRED (BIT #10)
004000          WRL=    4000            ;WRITE LOCK (BIT #11)
010000          MOL=    10000           ;MEDIUM ON-LINE (BIT #12)
020000          PIP=    20000           ;POSITIONING OPERATION IN PROGRESS (BIT #13)
040000          ERR=    40000           ;COMPOSITE ERROR (BIT #14)
100000          ATA=    100000          ;ATTENTION ACTIVE (BIT #15)

                        ;ERROR REGISTER #01 (RMER1) (#02)

000001          ILF=    1               ;ILLEGAL FUNCTION (BIT #0)
000002          ILR=    2               ;ILLEGAL REGISTER (BIT #1)
000004          RMR=    4               ;REGISTER MODIFICATION REFUSED (BIT #2)
000010          PAR=    10              ;PARITY ERROR (BIT #3)
000020          FER=    20              ;FORMAT ERROR (BIT #4)
000040          WCF=    40              ;WRITE CLOCK FAIL (BIT #5)
000100          ECH=    100             ;ECC HARD ERROR (BIT #6)
000200          HCE=    200             ;HEADER COMPARE ERROR (BIT #7)
000400          HCRC=   400             ;HEADER CRC ERROR (BIT #8)
001000          AOE=    1000            ;ADDRESS OVERFLOW ERROR (BIT #9)
002000          IAE=    2000            ;INVALID ADDRESS ERROR (BIT #10)
004000          WLE=    4000            ;WRITE LOCK ERROR (BIT #11)
010000          CTE=    10000           ;DRIVE TIMING ERROR (BIT #12)
020000          OPI=    20000           ;OPERATION INCOMPLETE (BIT #13)
040000          UNS=    40000           ;DRIVE UNSAFE (BIT #14)
100000          DCK=    100000          ;DATA CHECK ERROR (BIT 15)

                        ;MAINTAINABILITY REGISTER (RMMR1)(#03)

000001          DMD=    1               ;DIAGINOSTIC MODE (BIT #0)
000002          MCLK=   2               ;MAINTAINABILITY CLOCK (BIT #1)
000004          MINX=   4               ;MAINTAINABILITY INDEX (BIT #2)
000010          MSTCK=  10              ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
000020          MRD=    20              ;MAINTAINABILITY READ (BIT #4)
000040          MWR=    40              ;MAINTAINABILITY WRITE (BIT #5)
000200          DTSY=   200             ;MAINTAINABILITY SYNC DETECTED (BIT #7)

                        ;ATTENTION SUMMARY PSEUDO-REGISTER (RMAS) (#04)

000001          AT0=    1               ;DEVICE 0 (BIT #0)
000002          AT1=    2               ;DEVICE 1 (BIT #1)
000004          AT2=    4               ;DEVICE 2 (BIT #2)
000010          AT3=    10              ;DEVICE 3 (BIT #3)
000020          AT4=    20              ;DEVICE 4 (BIT #4)
000040          AT5=    40              ;DEVICE 5 (BIT #5)
000100          AT6=    100             ;DEVICE 6 (BIT #6)
000200          AT7=    200             ;DEVICE 7 (BIT #7)

                        ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RMDA) (#05)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)
```

```
                                ;DRIVE TYPE REGISTER (RMDT) (#06)

000001                  DT00=   1               ;DRIVE TYPE NUMBER BIT 1
000002                  DT01=   2               ;DRIVE TYPE NUMBER BIT 2
000004                  DT02=   4               ;DRIVE TYPE NUMBER BIT 3
000010                  DT03=   10              ;DRIVE TYPE NUMBER BIT 4
000020                  DT04=   20              ;DRIVE TYPE NUMBER BIT 5
000040                  DT05=   40              ;DRIVE TYPE NUMBER BIT 6
000100                  DT06=   100             ;DRIVE TYPE NUMBER BIT 7
000200                  DT07=   200             ;DRIVE TYPE NUMBER BIT 8
000400                  DT08=   400             ;DRIVE TYPE NUMBER BIT 9
004000                  DRQ=    4000            ;DRIVE REQUEST REQUIRED (BIT #11)
020000                  MOH=    20000           ;MOVING HEAD (BIT #13)
040000                  TAP=    40000           ;TAPE DRIVE (BIT #14)
100000                  NBA=    100000          ;NOT BLOCK ADDRESSED (BIT #15)

                                ;LOOK-AHEAD REGISTER (RMLA) (#07)

000100                  SC01=   100             ;SECTOR COUNT FIELD 0 (BIT #6)
000200                  SC02=   200             ;SECTOR COUNT FIELD 1 (BIT #7)
000400                  SC04=   400             ;SECTOR COUNT FIELD 2 (BIT #8)
001000                  SC10=   1000            ;SECTOR COUNT FIELD 3 (BIT #9)
002000                  SC20=   2000            ;SECTOR COUNT FIELD 4 (BIT #10)
004000                  TRK1=   4000            ;TRACK FIELD 1 (BIT #11)
010000                  TRK2=   10000           ;TRACK FIELD 2 (BIT #12)
020000                  TRK4=   20000           ;TRACK FIELD 3 (BIT #13)
040000                  TRK10=  40000           ;TRACK FIELD 4 (BIT #14)
100000                  TRK20=  100000          ;TRACK FIELD 5 (BIT #15)

                                ;OFFSET REGISTER (RMOF) (#11)

000001                  OFDIR=  1               ;OFFSET DIRECTION FOR RM01
002000                  HCI=    2000            ;HEADER COMPARE INHIBIT (BIT #10)
004000                  ECI=    4000            ;ERROR CORRECTION CODE INHIBIT (BIT #11
010000                  FMT16=  10000           ;FORMAT BIT (BIT #12

                                ;DESIRED CYLINDER ADDRESS (RMDC) (#12)
                                ;(EACH BIT IS CALLED BY BIT NUMBER)

                                ;SERIAL NUMBER REGISTER (RMSN) (#14)
                                ;(EACH IS CALLED BY BIT NUMBER)

                                ;RM01 MAINTENANCE  REGISTER #02 (RMMR2) (#15)

040000                  SKI=    40000           ;SEEK INCOMPLETE (BIT #14)

                                ;ECC POSITION REGISTER (RMEC1) (#16)
                                ;(EACH BIT IS CALLED BY BIT NUMBER)

                                ;ECC PATTERN REGISTER (RMEC2) (#17)
                                ;(EACH BIT IS CALLED BY BIT NUMBER)

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

                                .SBTTL  RM01 DRIVER COMMANDS
```

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

000101                  RNOP    =       101             ;NO OPERATION
000105                  SEEK    =       105             ;SEEK
000107                  RECAL   =       107             ;RECALIBRATE
000111                  DRVCLR  =       111             ;DRIVE CLEAR
000113                  RELSE   =       113             ;RELEASE
000115                  OFFSET  =       115             ;OFFSET
000117                  RTC     =       117             ;RETURN TO CENTER LINE
000121                  READIN  =       121             ;READ IN PRESET
000123                  ACK     =       123             ;PACK ACKNOWLEDGE
000131                  SEARCH  =       131             ;SEARCH
000141                  GETREG  =       141             ;GET REGISTERS
000143                  SETFMT  =       143             ;SET FORMAT (& ECI OR HCI)
000145                  SELDRV  =       145             ;SELECT DRIVE
000151                  WCKD    =       151             ;WRITE CHECK DATA
000153                  WCKHD   =       153             ;WRITE CHECK HEADER & DATA
000161                  WRTDAT  =       161             ;WRITE DATA
000163                  WRTHD   =       163             ;WRITE HEADER & DATA
000171                  RDDAT   =       171             ;READ DATA
000173                  RDHD    =       173             ;READ HEADER & DATA
```

```
                                .SBTTL  COMMON TAGS

                                ;;*****************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                    001100              .=1100
001100                          $CMTAG:                     ;;START OF COMMON TAGS
001100  000000      $PASS:  .WORD   0                       ;;CONTAINS PASS COUNT
001102  000         $TSTNM: .BYTE   0                       ;;CONTAINS THE TEST NUMBER
001103  000         $ERFLG: .BYTE   0                       ;;CONTAINS ERROR FLAG
001104  000000      $ICNT:  .WORD   0                       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000      $LPADR: .WORD   0                       ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000      $LPERR: .WORD   0                       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000      $ERTTL: .WORD   0                       ;;CONTAINS TOTAL ERRORS DETECTED
001114  000         $ITEMB: .BYTE   0                       ;;CONTAINS ITEM CONTROL BYTE
001115  001         $ERMAX: .BYTE   1                       ;;CONTAINS MAX. ERRORS PER TEST
001116  000000      $ERRPC: .WORD   0                       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000      $GDADR: .WORD   0                       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000      $BDADR: .WORD   0                       ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000      $GDDAT: .WORD   0                       ;;CONTAINS 'GOOD' DATA
001126  000000      $BDDAT: .WORD   0                       ;;CONTAINS 'BAD' DATA
001130  000000              .WORD   0                       ;;RESERVED--NOT TO BE USED
001132  000000              .WORD   0
001134  000         $AUTOB: .BYTE   0                       ;;AUTOMATIC MODE INDICATOR
001135  000         $INTAG: .BYTE   0                       ;;INTERRUPT MODE INDICATOR
001136  000000              .WORD   0
001140  177570      SWR:    .WORD   DSWR                    ;;ADDRESS OF SWITCH REGISTER
001142  177570      DISPLAY: .WORD  DDISP                   ;;ADDRESS OF DISPLAY REGISTER
001144  177560      $TKS:   177560                          ;;TTY KBD STATUS
001146  177562      $TKB:   177562                          ;;TTY KBD BUFFER
001150  177564      $TPS:   177564                          ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566      $TPB:   177566                          ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000         $NULL:  .BYTE   0                       ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002         $FILLS: .BYTE   2                       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012         $FILLC: .BYTE   12                      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000         $TPFLG: .BYTE   0                       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000      $ESCAPE:0                               ;;ESCAPE ON ERROR ADDRESS
001162  177607  000377  $BELL:  .ASCIZ  <207><377><377>    ;;CODE FOR BELL
001166  077         $QUES:  .ASCII  /?/                     ;;QUESTION MARK
001167  015         $CRLF:  .ASCII  <15>                    ;;CARRIAGE RETURN
001170  000012      $LF:    .ASCIZ  <12>                    ;;LINE FEED
                                ;;*****************************************************************
                    000015      CR      =       15
                    000012      LF      =       12
001172  176700      $RMADR: .WORD   176700          ;RH11/RM01 UNIBUS ADDRESS
001174  000254      $RMVEC: .WORD   254             ;RH11 INTERRUPT VECTOR
001176  172540      $LKCSR: .WORD   172540          ;ADDRESS OF KW11-P CSR
001200  172542      $LKCSB: .WORD   172542          ;ADDRESS OF KW11-P COUNTER BUFFER
001202  000104  000106  $LPVEC: .WORD   104,106     ;ADDRESS OF KW11-P VECTOR
001206  177546      $LKS:   .WORD   177546          ;ADDRESS OF KW11-L CONTROL REGISTER
001210  000100  000102  $LLVEC: .WORD   100,102     ;ADDRESS OF KW11-L VECTOR
001214  000000      DRIVE:  .WORD   0               ;CONTAINS DRIVE NUMBER SELECTED
001216  000000      SOFSW:  .WORD   0               ;CONTENTS ARE FOR SOFTWARE DECISIONS
001220  000000      MODE:   .WORD   0               ;'FORMAT & VERIFY' OR 'CHECK' MODE INDICATOR
001222  001466      ENDCYL: .WORD   822.            ;ENDING CYLINDER
```

```
898  001224  000000  BEGCYL: .WORD   0          ;STARTING CYLINDER
899  001226  000004  ENDTRK: .WORD   4.         ;ENDING TRACK
900  001230  000000  BEGTRK: .WORD   0          ;STARTING TRACK
901  001232  000000  TTRKS:  .WORD   0          ;TOTAL # OF TRACKS TO BE FORMATTED
902  001234  000000  TTRKSC: .WORD   0          ;TOTAL # OF TRACKS COUNTER
903  001236  000000  TRKCNT: .WORD   0          ;COUNTS TRKS FROM 0-5 PER CYL
904  001240  000000  PATSEL: .WORD   0          ;CONTAINS PATTERN SELECTED
905  001242  000000  PATA:   .WORD   0          ;1ST WORD OF PATTERN
906  001244  000000  PATB:   .WORD   0          ;2ND WORD OF PATTERN
907  001246  000000  CYLCK:  .WORD   0          ;STORE CYLINDER ADDRESS FOR FORMAT VERIFICATION
908  001250  000000  RETRY:  .WORD   0          ;MAINTAINS # OF WRITE RETRIES MADE
909  001252  000000  SAVSEC: .WORD   0          ;CONTAINS LAST BAD SECTOR ON FORMAT
910  001254  000000  SAVWC:  .WORD   0          ;CONTAINS WC FOR REMAINING SECTORS ON ERROR
911  001256  000000  WC:     .WORD   0          ;30 OR 32 SECTOR TRACK SIZE (IN WORDS)
912  001260  000000  MWC:    .WORD   0          ;2'S COMPLEMENT OF 'WC'
913  001262  000000  HEDERR: .WORD   0          ;POSITIONING ERROR DURING FORMAT INDICATOR
914  001264  000000  SEC30:  .WORD   0          ;30 OR 32 SECTOR MODE INDICATOR
915                                             ;0 = 30 SECTOR MODE
916                                             ;1'S = 32 SECTOR MODE
917  001266  000000  MAXSEC: .WORD   0          ;MAXIMUM SECTOR ADDRESS (FOR EITHER 30 OR 32 SECTOR
918                                             ;FORMAT)
919  001270  000000  DS.CYL: .WORD   0          ;ADDRESS OF CURRENT CYLINDER
920  001272  000000  DS.TRK: .WORD   0          ;ADDRESS OF CURRENT TRACK
921  001274  000000  DDRIVE: .WORD   0          ;DRIVE NUMBER OF 'DRIVER' ERROR MESSAGES
922  001276  000000  ATTN:   .WORD   0          ;ATTENTION REGISTER IMAGE FOR 'DRIVER'
923                                             ;ERROR MESSAGES
924  001300  000000  CNTLC:  .WORD   0          ;ADDRESS OF '^C' RETURN
925  001302  000000  CHGADR: .WORD   0          ;'CHANGE RH11 ADDRESS' FLAG
926  001304  000000  WRAP:   .WORD   0          ;WRAP AROUND FLAG,FORMAT FROM HIGH
927                                             ;CYLINDER # TO LOW CYLINDER #
928
929                 ;RH11/RM01 REGISTERS STORED HERE AFTER AN OPERATION
930
931  001306  000000  RM.REG:  .WORD   0          ;RMCS1
932  001310  000000          .WORD   0          ;RMWC
933  001312  000000          .WORD   0          ;RMBA
934  001314  000000          .WORD   0          ;RMDA
935  001316  000000          .WORD   0          ;RMCS2
936  001320  000000          .WORD   0          ;RMDS
937  001322  000000          .WORD   0          ;RMER1
938  001324  000000          .WORD   0          ;RMAS
939  001326  000000          .WORD   0          ;RMLA
940  001330  000000          .WORD   0          ;RMDB
941  001332  000000          .WORD   0          ;RMMR1
942  001334  000000          .WORD   0          ;RMDT
943  001336  000000          .WORD   0          ;RMSN
944  001340  000000          .WORD   0          ;RMOF
945  001342  000000          .WORD   0          ;RMCA
946  001344  000000          .WORD   0          ;RMDC
947  001346  000000          .WORD   0          ;RMER2
948  001350  000000          .WORD   0          ;RMMR2
949  001352  000000          .WORD   0          ;RMEC1
950  001354  000000          .WORD   0          ;RMEC2
951
952
953                 ;DATA/PARAMETER BLOCK - USED FOR ALL DRIVE OPERATION
```

```
                                FMTDPB: .BYTE   0               ;DRIVE NUMBER
                                        .BYTE   0               ;OFFSET VALUE OR FMT22,ECI, AND HCI
                                        .BYTE   0               ;COMMAND
                                        .BYTE   0               ;PSEL AND A17 AND A16
                                        .WORD   0               ;WORD COUNT (NEG)
                                        .WORD   0               ;BUFFER ADDRESS
                                        .BYTE   0               ;SECTOR ADDRESS
                                        .BYTE   0               ;TRACK ADDRESS
                                        .WORD   0               ;CYLINDER ADDRESS
                                        .WORD   RM.REG          ;ERROR TABLE POINTER
                                        .WORD   0               ;STATUS-ERROR INDICATOR
                                                                ;BIT 15 = 1: ERROR OCCURRED
                                                                ;BIT 07 = 1: DONE
                                                                ;BIT 14-10 AND BIT 06-03
                                                                ;INDICATE TYPE OF ERROR


                                ;PARAMETER POINTER TABLE

001376  001430  001467  001224  TABLE:  PAR1,823.,BEGCYL
001404  001443  000005  001230          PAR2,5.,BEGTRK
001412  001456  001467  001222          PAR3,823.,ENDCYL
001420  001467  000005  001226          PAR4,5.,ENDTRK,0
001426  000000

                                ;ASCII MESSAGES FOR ADDRESS PARAMETERS

001430  052123  051101  020124  PAR1:   .ASCIZ  @START CYL @
001436  054503  020114     000
001443     123  040524  052122  PAR2:   .ASCIZ  @START TRK @
001450  052040  045522  000040
001456  047105  020104  054503  PAR3:   .ASCIZ  @END CYL @
001464  020114     000
001467     105  042116  052040  PAR4:   .ASCIZ  @END TRK @
001474  045522  000040

                                ;SECTOR BUFFER ADDRESS TABLE

001500  025732                  ADRTBL: .WORD   BUFP             ;ADDRESS OF SECTOR 0
001502  026736                          .WORD   BUFP+<516.*1.>   ;ADDRESS OF SECTOR 1
001504  027742                          .WORD   BUFP+<516.*2.>   ;ADDRESS OF SECTOR 2
001506  030746                          .WORD   BUFP+<516.*3.>   ;ADDRESS OF SECTOR 3
001510  031752                          .WORD   BUFP+<516.*4.>   ;ADDRESS OF SECTOR 4
001512  032756                          .WORD   BUFP+<516.*5.>   ;ADDRESS OF SECTOR 5
001514  033762                          .WORD   BUFP+<516.*6.>   ;ADDRESS OF SECTOR 6
001516  034766                          .WORD   BUFP+<516.*7.>   ;ADDRESS OF SECTOR 7
001520  035772                          .WORD   BUFP+<516.*8.>   ;ADDRESS OF SECTOR 8
001522  036776                          .WORD   BUFP+<516.*9.>   ;ADDRESS OF SECTOR 9
001524  040002                          .WORD   BUFP+<516.*10.>  ;ADDRESS OF SECTOR 10
001526  041006                          .WORD   BUFP+<516.*11.>  ;ADDRESS OF SECTOR 11
001530  042012                          .WORD   BUFP+<516.*12.>  ;ADDRESS OF SECTOR 12
001532  043016                          .WORD   BUFP+<516.*13.>  ;ADDRESS OF SECTOR 13
001534  044022                          .WORD   BUFP+<516.*14.>  ;ADDRESS OF SECTOR 14
001536  045026                          .WORD   BUFP+<516.*15.>  ;ADDRESS OF SECTOR 15
001540  046032                          .WORD   BUFP+<516.*16.>  ;ADDRESS OF SECTOR 16
```

```
1010  001542  047036                  .WORD   BUFP+<516.*17.>  ;ADDRESS OF SECTOR 17
1011  001544  050042                  .WORD   BUFP+<516.*18.>  ;ADDRESS OF SECTOR 18
1012  001546  051046                  .WORD   BUFP+<516.*19.>  ;ADDRESS OF SECTOR 19
1013  001550  052052                  .WORD   BUFP+<516.*20.>  ;ADDRESS OF SECTOR 20
1014  001552  053056                  .WORD   BUFP+<516.*21.>  ;ADDRESS OF SECTOR 21
1015  001554  054062                  .WORD   BUFP+<516.*22.>  ;ADDRESS OF SECTOR 22
1016  001556  055066                  .WORD   BUFP+<516.*23.>  ;ADDRESS OF SECTOR 23
1017  001560  056072                  .WORD   BUFP+<516.*24.>  ;ADDRESS OF SECTOR 24
1018  001562  057076                  .WORD   BUFP+<516.*25.>  ;ADDRESS OF SECTOR 25
1019  001564  060102                  .WORD   BUFP+<516.*26.>  ;ADDRESS OF SECTOR 26
1020  001566  061106                  .WORD   BUFP+<516.*27.>  ;ADDRESS OF SECTOR 27
1021  001570  062112                  .WORD   BUFP+<516.*28.>  ;ADDRESS OF SECTOR 28
1022  001572  063116                  .WORD   BUFP+<516.*29.>  ;ADDRESS OF SECTOR 29
1023  001574  064122                  .WORD   BUFP+<516.*30.>  ;ADDRESS OF SECTOR 30
1024  001576  065126                  .WORD   BUFP+<516.*31.>  ;ADDRESS OF SECTOR 31
1025
1026                          ;REMAINING WORD COUNT TABLE
1027
1028  001600  000402          WCTBL:  .WORD   258.             ;REMAINING WORD COUNT AFTER SECTOR 0
1029  001602  001004                  .WORD   258.+<258.*1.>   ;REMAINING WORD COUNT AFTER SECTOR 1
1030  001604  001406                  .WORD   258.+<258.*2.>   ;REMAINING WORD COUNT AFTER SECTOR 2
1031  001606  002010                  .WORD   258.+<258.*3.>   ;REMAINING WORD COUNT AFTER SECTOR 3
1032  001610  002412                  .WORD   258.+<258.*4.>   ;REMAINING WORD COUNT AFTER SECTOR 4
1033  001612  003014                  .WORD   258.+<258.*5.>   ;REMAINING WORD COUNT AFTER SECTOR 5
1034  001614  003416                  .WORD   258.+<258.*6.>   ;REMAINING WORD COUNT AFTER SECTOR 6
1035  001616  004020                  .WORD   258.+<258.*7.>   ;REMAINING WORD COUNT AFTER SECTOR 7
1036  001620  004422                  .WORD   258.+<258.*8.>   ;REMAINING WORD COUNT AFTER SECTOR 8
1037  001622  005024                  .WORD   258.+<258.*9.>   ;REMAINING WORD COUNT AFTER SECTOR 9
1038  001624  005426                  .WORD   258.+<258.*10.>  ;REMAINING WORD COUNT AFTER SECTOR 10
1039  001626  006030                  .WORD   258.+<258.*11.>  ;REMAINING WORD COUNT AFTER SECTOR 11
1040  001630  006432                  .WORD   258.+<258.*12.>  ;REMAINING WORD COUNT AFTER SECTOR 12
1041  001632  007034                  .WORD   258.+<258.*13.>  ;REMAINING WORD COUNT AFTER SECTOR 13
1042  001634  007436                  .WORD   258.+<258.*14.>  ;REMAINING WORD COUNT AFTER SECTOR 14
1043  001636  010040                  .WORD   258.+<258.*15.>  ;REMAINING WORD COUNT AFTER SECTOR 15
1044  001640  010442                  .WORD   258.+<258.*16.>  ;REMAINING WORD COUNT AFTER SECTOR 16
1045  001642  011044                  .WORD   258.+<258.*17.>  ;REMAINING WORD COUNT AFTER SECTOR 17
1046  001644  011446                  .WORD   258.+<258.*18.>  ;REMAINING WORD COUNT AFTER SECTOR 18
1047  001646  012050                  .WORD   258.+<258.*19.>  ;REMAINING WORD COUNT AFTER SECTOR 19
1048  001650  012452                  .WORD   258.+<258.*20.>  ;REMAINING WORD COUNT AFTER SECTOR 20
1049  001652  013054                  .WORD   258.+<258.*21.>  ;REMAINING WORD COUNT AFTER SECTOR 21
1050  001654  013456                  .WORD   258.+<258.*22.>  ;REMAINING WORD COUNT AFTER SECTOR 22
1051  001656  014060                  .WORD   258.+<258.*23.>  ;REMAINING WORD COUNT AFTER SECTOR 23
1052  001660  014462                  .WORD   258.+<258.*24.>  ;REMAINING WORD COUNT AFTER SECTOR 24
1053  001662  015064                  .WORD   258.+<258.*25.>  ;REMAINING WORD COUNT AFTER SECTOR 25
1054  001664  015466                  .WORD   258.+<258.*26.>  ;REMAINING WORD COUNT AFTER SECTOR 26
1055  001666  016070                  .WORD   258.+<258.*27.>  ;REMAINING WORD COUNT AFTER SECTOR 27
1056  001670  016472                  .WORD   258.+<258.*28.>  ;REMAINING WORD COUNT AFTER SECTOR 28
1057  001672  017074                  .WORD   258.+<258.*29.>  ;REMAINING WORD COUNT AFTER SECTOR 29
1058  001674  017476                  .WORD   258.+<258.*30.>  ;REMAINING WORD COUNT AFTER SECTOR 30
1059  001676  020100                  .WORD   258.+<258.*31.>  ;REMAINING WORD COUNT AFTER SECTOR 31
1060
1061                          .EVEN
1062
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001700                          $ERRTA:
                                ;ERROR 1

001700  022605                          EM1                     ;RH11 INTERRUPT OCCURRED (RMAS=0)
001702  023724                          DH1
001704  025256                          DT1
001706  025612                          DF1

                                ;ERROR 2

001710  022646                          EM2                     ;UNEXPECTED ATTENTION OCCURRED
001712  023772                          DH2
001714  025274                          DT2
001716  025616                          DF2

                                ;ERROR 3

001720  022704                          EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
001722  024044                          DH3
001724  025310                          DT3
001726  025622                          DF3

                                ;ERROR 4

001730  022742                          EM4                     ;MASSBUS PARITY ERROR (PAR=1)
001732  024115                          DH4
001734  025324                          DT4
001736  025626                          DF4

                                ;ERROR 5

001740  000000                          0                       ;UNUSED
001742  000000                          0
001744  000000                          0
001746  000000                          0

                                ;ERROR 6

001750  023033                          EM6                     ;RH11 DIDN'T RESPOND TO ADDRESSING
001752  024200                          DH6
001754  025342                          DT6
001756  025632                          DF6
```

```
1119
1120                                    ;ERROR 7
1121
1122  001760  000000                            0                       ;UNUSED
1123  001762  000000                            0
1124  001764  000000                            0
1125  001766  000000                            0
1126
1127                                    ;ERROR 10
1128
1129  001770  023075                            EM10                    ;DRIVE OFFLINE
1130  001772  024213                            DH10
1131  001774  025344                            DT10
1132  001776  025636                            DF10
1133
1134                                    ;ERROR 11
1135
1136  002000  023113                            EM11                    ;PERSISTENT DRIVE UNSAFE ERROR
1137  002002  024213                            DH10
1138  002004  025344                            DT10
1139  002006  025636                            DF10
1140
1141                                    ;ERROR 12
1142
1143  002010  023151                            EM12                    ;UNCORRECTABLE MASSBUS PARITY ERROR
1144  002012  024213                            DH10
1145  002014  025344                            DT10
1146  002016  025636                            DF10
1147
1148                                    ;ERROR 13
1149
1150  002020  023214                            EM13                    ;SOFTWARE TIMEOUT
1151  002022  024213                            DH10
1152  002024  025344                            DT10
1153  002026  025636                            DF10
1154
1155                                    ;ERROR 14
1156
1157  002030  023235                            EM14                    ;DRIVE UNSAFE ERROR
1158  002032  024213                            DH10
1159  002034  025344                            DT10
1160  002036  025636                            DF10
1161
1162                                    ;ERROR 15
1163
1164  002040  023260                            EM15                    ;CONTROLLER/DRIVE ERROR DURING WRITE
1165  002042  024213                            DH10
1166  002044  025344                            DT10
1167  002046  025636                            DF10
1168
1169                                    ;ERROR 16
1170
1171  002050  023324                            EM16                    ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1172  002052  024213                            DH10
1173  002054  025344                            DT10
1174  002056  025636                            DF10
```

```
1175
1176                                    ;ERROR 17
1177
1178  002060  023376                            EM17            ;RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE
1179  002062  024441                            DH17
1180  002064  025416                            DT17
1181  002066  025652                            DF17
1182
1183                                    ;ERROR 20
1184
1185  002070  023454                            EM20            ;DATA ERROR DURING WRITE CHECK
1186  002072  024510                            DH20
1187  002074  025430                            DT20
1188  002076  025656                            DF20
1189
1190                                    ;ERROR 21
1191
1192  002100  023512                            EM21            ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
1193  002102  024213                            DH10
1194  002104  025344                            DT10
1195  002106  025636                            DF10
1196
1197                                    ;ERROR 22
1198
1199  002110  023563                            EM22            ;'HCE' ERROR VERIFYING HEADERS
1200  002112  024213                            DH10
1201  002114  025344                            DT10
1202  002116  025636                            DF10
1203
1204                                    ;ERROR 23
1205
1206  002120  023632                            EM23            ;CYLINDER FIELD IN HEADER IS NOT CORRECT
1207  002122  025005                            DH23
1208  002124  025512                            DT23
1209  002126  025676                            DF23
1210
1211                                    ;ERROR 24
1212
1213  002130  023702                            EM24            ;WRITE CHECK ERROR
1214  002132  025103                            DH24
1215  002134  025524                            DT24
1216  002136  025702                            DF24
1217
1218                                    ;;*****************************************************************
1219
1220                                    .SBTTL  MAIN PROGRAM
1221
1222                                    ;;*****************************************************************
1223
1224  002140  012737  177777  001302  BEGIN:  MOV     #-1,CHGADR      ;SET CHANGE 'RH70 BUS ADDRESS' INDICATOR
1225  002146  000402                          BR      BEGIN2          ;START THE PROGRAM
1226  002150  005037  001302          BEGIN1: CLR     CHGADR          ;CLEAR THE 'CHANGE RH70 ADDRESS' INDICATOR
1227  002154  000005                  BEGIN2: RESET                   ;CLEAR THE BUS
1228                                    .SBTTL  INITIALIZE THE COMMON TAGS
1229                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1230  002156  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
```

```
1231  002162  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1232  002164  022706  001140                  CMP     #SWR,R6 ;;DONE?
1233  002170  001374                          BNE     .-6             ;;LOOP BACK IF NO
1234  002172  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1235                          ;;INITIALIZE A FEW VECTORS
1236  002176  012737  007304  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1237  002204  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1238  002212  012737  012462  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1239  002220  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
1240  002226  005037  001160                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1241  002232  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1242                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1243                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1244  002240  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1245  002244  012737  002300  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1246  002252  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1247  002260  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1248  002266  022777  177777  176644          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1249  002274  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1250                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1251  002276  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1252  002300  012716  002306          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1253  002304  000002                          RTI
1254  002306  012737  000176  001140  65$.    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1255  002314  012737  000174  001142          MOV     #DISPREG,DISPLAY
1256  002322  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1257
1258  002326  000005                          RESET                   ;CLEAR WORLD
1259  002330  012737  000240  000036          MOV     #PR5,@#TRAPVEC+2  ;CHANGE TRAP PRIORITY BACK TO 5
1260  002336  012737  000240  000032          MOV     #PR5,@#EMTVEC+2 ;CHANGE EMT PRIORITY BACK TO 5
1261  002344  012737  002150  001300          MOV     #BEGIN1,CNTLC   ;'CONTROL C' ADDRESS
1262  002352  005227  177777                  INC     #-1             ;FIRST START ?
1263  002356  001010                          BNE     1$              ;BR IF NOT
1264  002360  104401  025732                  TYPE    ,TITLE          ;ADRS OF 'TITLE' MESSAGE
1265  002364  122737  000011  000041          CMPB    #11,41          ;LOADED FROM AN RMD1 ?
1266  002372  001002                          BNE     1$              ;BR IF NOT
1267  002374  104401  026010                  TYPE    ,LOADRV         ;INSTRUCT OPERATOR TO REMOVE 'XXDP'
1268                                                                  ;PACK FROM DRIVE 0
1269  002400  004737  010652          1$:     JSR     PC,$TKINT       ;TURN ON THE KEYBOARD INTERRUPT
1270                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1271  002404  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1272  002410  001006                          BNE     67$             ;;BRANCH IF YES
1273  002412  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1274  002420  001005                          BNE     68$             ;;BRANCH IF NO
1275  002422  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1276  002424  000403                          BR      68$
1277  002426  112737  000001  001134  67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1278  002434                          68$:
1279  002434  005227  177777                  INC     #-1             ;CHECK FOR FIRST START
1280  002440  001010                          BNE     SETVEC          ;BR IF NOT
1281  002442  004737  026226                  JSR     PC,BUSADR       ;CHECK THE RH11 ADDRESS
1282  002446  013737  001172  012714          MOV     $RMADR,RMADR    ;RH11 ADDRESS
1283  002454  013737  001174  012716          MOV     $RMVEC,RMVEC    ;RH11 VECTOR ADDRESS
1284
1285                          ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
1286                          ;       PROGRAM WILL USE
```

```
1287
1288  002462  004737  006352          SETVEC: JSR     PC,ST.CLK       ;START THE CLOCK
1289  002466  004737  012732                  JSR     PC,RMINIT       ;INITIALIZE THE RM01 DRIVER
1290  002472  012737  177777  012654          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
1291  002500  005227  177777                  INC     #-1             ;SEE IF FIRST START
1292  002504  001404                          BEQ     11$             ;BR IF YES
1293  002506  032777  000004  176424          BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
1294  002514  001076                          BNE     10$             ;BR IF NOT
1295  002516  012737  000340  177776  11$:    MOV     #PR7,PS         ;SET PRIORITY TO 7
1296  002524  005004                          CLR     R4              ;DRIVE TABLE POINTER
1297  002526  104401  001167                  TYPE    ,$CRLF          ;CR-LF
1298  002532  104401  021272                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
1299  002536                          1$:
1300  002536  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
1301                                                                  ;;TYPE DRIVE NUMBER
1302  002540  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1303  002542     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1304  002543     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1305  002544  104401  021335                  TYPE    ,LIN4SP         ;SPACES
1306  002550  105764  012566                  TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
1307  002554  100416                          BMI     4$              ;BR IF UNSAFE
1308  002556  001020                          BNE     5$              ;BR IF ONLINE
1309  002560  105764  012576                  TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
1310  002564  001404                          BEQ     2$              ;BR IF NONEXISTENT
1311  002566  100006                          BPL     3$              ;BR IF OFFLINE
1312  002570  104401  021211                  TYPE    ,NOTRM          ;DRIVE NOT AN RM01
1313  002574  000440                          BR      9$              ;CHECK NEXT DRIVE
1314  002576  104401  021226          2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
1315  002602  000435                          BR      9$              ;CHECK NEXT DRIVE
1316  002604  104401  021170          3$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
1317  002610  000405                          BR      6$              ;PRINT DRIVE TYPE
1318  002612  104401  021243          4$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
1319  002616  000402                          BR      6$              ;PRINT DRIVE TYPE
1320  002620  104401  021201          5$:     TYPE    ,UNTON          ;DRIVE ONLINE
1321  002624  104401  021337          6$:     TYPE    ,LINSP          ;SPACES
1322  002630  012737  021253  002674          MOV     #RM01B,8$       ;ADDRESS OF RP04 MESSAGE
1323  002636  132764  000001  012576          BITB    #BIT00,DRVTYP(R4)  ;RM01 ?
1324  002644  001012                          BNE     7$              ;BR IF YES
1325  002646  012737  021260  002674          MOV     #RP05,8$        ;ADDRESS OF RP05 MESSAGE
1326  002654  132764  000002  012576          BITB    #BIT01,DRVTYP(R4)  ;RP05 ?
1327  002662  001003                          BNE     7$              ;BR IF YES
1328  002664  012737  021265  002674          MOV     #RM01A,8$       ;ADDRESS OF RM01 MESSAGE
1329  002672  104401                  7$:     TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
1330  002674  000000                  8$:     .WORD   0               ;MESSAGE ADDRESS HERE
1331  002676  104401  001167          9$:     TYPE    ,$CRLF          ;CR-LF
1332  002702  005204                          INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
1333  002704  020427  000010                  CMP     R4,#8.          ;FINISHED ?
1334  002710  001312                          BNE     1$              ;BR IF NOT
1335  002712  104401  001167          10$:    TYPE    ,$CRLF          ;CR-LF
1336
1337
1338                                  ;SEE WHICH MODE THE PROGRAM IS TO BE RUN IN.
1339                                  ;MODES ARE: 'FORMAT & VERIFY' OR 'CHECK FORMAT'
1340
1341  002716  005037  001220          M1:     CLR     MODE            ;SET MODE TO 'CHECK FORMAT'
1342  002722  104401  021547                  TYPE    ,MMODE          ;TYPE 'PROGRAM MODE'
```

```
1343  002726  104411                          RDLIN                   ;READ THE KEYBOARD
1344  002730  012601                          MOV     (SP)+,R1        ;GET ADDRESS OF INPUT
1345  002732  105711                          TSTB    (R1)            ;'CR' ENTERED (DEFAULT)
1346  002734  001414                          BEQ     2$              ;BR IF YES
1347  002736  122711  000103                  CMPB    #'C,(R1)        ;'CHECK' ?
1348  002742  001406                          BEQ     1$              ;BR IF YES
1349  002744  122711  000106                  CMPB    #'F,(R1)        ;'FORMAT' ?
1350  002750  001411                          BEQ     3$              ;BR IF YES
1351  002752  104401  001166                  TYPE    ,$QUES          ;NO CORRECT ENTRY
1352  002756  000757                          BR      M1              ;TRY AGAIN
1353  002760  104401  021622          1$:     TYPE    ,MHECK          ;TYPE REST OF 'CHECK'
1354  002764  000410                          BR      M1A             ;GET STARTING ADDRESS
1355  002766  104401  021601          2$:     TYPE    ,MFORMT         ;TYPE DEFAULT MESSAGE
1356  002772  000402                          BR      4$              ;SET UP MODE
1357  002774  104401  021635          3$:     TYPE    ,MORMAT         ;TYPE REST OF 'FORMAT'
1358  003000  012737  177777  001220  4$:     MOV     #-1,MODE        ;SET MODE TO 'FORMAT & VERIFY'
1359
1360                                  ;FIND OUT IF FORMAT IS TO BE IN 30 OR 32 SECTOR MODE
1361
1362  003006  104401  021655          M1A:    TYPE    ,MSIZE          ;TYPE FORMAT MODE REQUEST
1363  003012  104411                          RDLIN                   ;READ THE KEYBOARD
1364  003014  012601                          MOV     (SP)+,R1        ;ADDRESS OF ENTRY
1365  003016  122711  000131                  CMPB    #'Y,(R1)        ;IS ENTRY A 'Y' ?
1366  003022  001410                          BEQ     1$              ;BR IF IT IS
1367  003024  122711  000116                  CMPB    #'N,(R1)        ;IS ENTRY A 'N' ?
1368  003030  001424                          BEQ     2$              ;BR IF IT IS
1369  003032  105711                          TSTB    (R1)            ;DEFAULT MODE ?
1370  003034  001403                          BEQ     1$              ;BR IF IT IS
1371  003036  104401  001166                  TYPE    ,$QUES          ;TYPE '?'
1372  003042  000761                          BR      M1A             ;TRY AGAIN
1373  003044  104401  021726          1$:     TYPE    ,MSEC22         ;TYPEOUT MODE SELECTED
1374  003050  012737  020100  001256          MOV     #<258.*32.>,WC  ;TRACK SIZE IN 32 SECTOR MODE
1375  003056  012737  157700  001260          MOV     #-<258.*32.>,MWC  ;2'S COMPLEMENT WORD COUNT
1376  003064  012737  177777  001264          MOV     #-1,SEC30       ;32 SECTOR INDICATOR
1377  003072  012737  000037  001266          MOV     #31.,MAXSEC     ;MAX SECTOR ADDRESS IN 32 SECTOR MODE
1378  003100  000415                          BR      M1B             ;CONTINUE
1379  003102  104401  022005          2$:     TYPE    ,MSEC20         ;TYPE OUT 30 SECTOR MODE SELECTED
1380  003106  012737  017074  001256          MOV     #<258.*30.>,WC  ;TRACK SIZE IN 30 SECTOR MODE
1381  003114  012737  160704  001260          MOV     #-<258.*30.>,MWC  ;2'S COMPLEMENT WORD COUNT
1382  003122  005037  001264                  CLR     SEC30           ;30 SECTOR INDICATOR
1383  003126  012737  000035  001266          MOV     #29.,MAXSEC     ;MAX SECTOR ADDRESS IN 30 SECTOR MODE
1384  003134  005037  001230          M1B:    CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
1385  003140  005037  001224                  CLR     BEGCYL          ;CLEAR BEGINNING CYLINDER ADDRESS
1386  003144  005037  001304                  CLR     WRAP            ;CLEAR WRAP AROUND FLAG
1387  003150  012737  000004  001226          MOV     #4.,ENDTRK      ;SETUP END TRACK ADDRESS
1388  003156  012737  000002  001240          MOV     #2,PATSEL       ;SETUP FOR WORST CASE PATTERN
1389  003164  112737  177777  001356          MOVB    #-1,FMTDPB      ;SETUP INVALID DRIVE NUMBER
1390  003172  000400                          BR      MC              ;BRANCH TO START
1391
1392                                  ;GO FIND OUT WHAT DRIVE
1393
1394  003174  012737  006526  001300  MO:     MOV     #OENTER,CNTLC   ;CONTROL C ABORT ENTRANCE
1395  003202  005037  001112                  CLR     $ERTTL          ;CLEAR THE ERROR ACCUMULATOR
1396  003206  004737  010652                  JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
1397  003212  104401  021311                  TYPE    ,MUNIT          ;ASK FOR DRIVE NUMBER
1398  003216  104411                          RDLIN                   ;READ THE KEYBOARD
```

```
 1399  003220  012601                          MOV     (SP)+,R1        ;ADDRESS OF ENTRY
 1400  003222  004537  007074                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 1401  003226  003252                          3$                      ;ILLEGAL CHARACTER
 1402  003230  003242                          2$                      ;CARRIAGE RETURN
 1403  003232  003252                          3$                      ;" "
 1404  003234  003242                          2$                      ;"'"
 1405  003236  003260                          4$                      ;DIGIT 0-7
 1406  003240  003252                          3$                      ;DIGIT 8-9
 1407  003242  005002                  2$:     CLR     R2              ;SELECT DRIVE ZERO
 1408  003244  104401  021333                  TYPE    ,MDRVD          ;GO TYPE DEFAULT DRIVE NUMBER
 1409  003250  000403                          BR      4$              ;GO SEE IF DRIVE ZERO IS THERE
 1410  003252  104401  001166          3$:     TYPE    ,$QUES          ;TYPE '?'
 1411  003256  000746                          BR      M0              ;ASK FOR DRIVE NUMBER AGAIN
 1412  003260  010237  001214          4$:     MOV     R2,DRIVE        ;SAVE DRIVE NUMBER
 1413  003264  005702                          TST     R2              ;SEE IF DRIVE 0
 1414  003266  001004                          BNE     5$              ;BR IF NOT
 1415  003270  122737  000011  000041          CMPB    #11,41          ;PROGRAM LOADED FROM AN RM01 ?
 1416  003276  001431                          BEQ     7$              ;BR IF IT WAS, CAN'T FORMAT THE DRIVE
 1417  003300  004737  006352          5$:     JSR     PC,ST.CLK       ;START THE CLOCK
 1418  003304  004737  012732                  JSR     PC,RMINIT       ;GO SEE WHAT DRIVES ARE AVAILABLE
 1419  003310  012737  177777  012654          MOV     #-1,SAVEFG      ;SAVE THE REGISTERS
 1420  003316  012737  177777  012656          MOV     #-1,SEEKFG      ;SET 'NO OPTIMIZATION' FLAG
 1421  003324  005037  177776                  CLR     PS              ;SET PRIORITY BACK TO ZERO
 1422  003330  105762  012566                  TSTB    DRVSTA(R2)      ;LOOK AT DRIVE STATUS
 1423  003334  003015                          BGT     8$              ;BRANCH IF ONLINE
 1424  003336  001403                          BEQ     6$              ;BR IF DRIVE NOT AVAILABLE
 1425  003340  104401  021515                  TYPE    ,MUSDR          ;'DRIVE UNSAFE'
 1426  003344  000713                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
 1427  003346  105762  012576          6$:     TSTB    DRVTYP(R2)      ;A DRIVE PRESENT?
 1428  003352  001003                          BNE     7$              ;BR IF SO
 1429  003354  104401  021414                  TYPE    ,MDRNP          ;TYPE 'DRIVE NOT PRESENT'
 1430  003360  000705                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
 1431  003362  104401  021441          7$:     TYPE    ,MER11          ;'DRIVE NOT AVAILABLE'
 1432  003366  000702                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
 1433  003370  123737  001214  001356  8$:     CMPB    DRIVE,FMTDPB    ;SAME DRIVE AS LAST TIME ?
 1434  003376  001422                          BEQ     M2              ;BR IF IT IS
 1435  003400  113737  001214  001356          MOVB    DRIVE,FMTDPB    ;SETUP DRIVE ADDRESS
 1436  003406  012737  000632  001222          MOV     #410.,ENDCYL    ;SETUP FOR RM01
 1437  003414  132762  000003  012576          BITB    #BIT00!BIT01,DRVTYP(R2)  ;SEE IF DRIVE RM01
 1438  003422  001003                          BNE     9$              ;BR IF EITHER
 1439  003424  012737  001466  001222          MOV     #822.,ENDCYL    ;SETUP ENDING CYLINDER FOR RM01
 1440  003432  005037  001230          9$:     CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
 1441  003436  005037  001224                  CLR     BEGCYL          ;CLEAR STARTING CYLINDER ADDRESS
 1442  003442  000400                          BR      M2              ;GET ADDRESS LIMITS FROM THE OPERATOR
 1443
 1444                                  ;GET ADDRESS LIMITS
 1445
 1446  003444  104401  021342          M2:     TYPE    ,ENTADR         ;'ENTER ADDRESS LIMITS'
 1447  003450  004737  006704                  JSR     PC,PARENT       ;GET THE ADDRESS LIMITS
 1448  003454  005037  001304                  CLR     WRAP            ;CLEAR WRAP FLAG
 1449  003460  023737  001222  001224          CMP     ENDCYL,BEGCYL   ;SEE IF ENDING CYLINDER EQ TO GT THAN BEGINNING
 1450  003466  001402                          BEQ     2$              ;BR IF ON THE SAME CYLINDER
 1451  003470  103410                          BLO     4$              ;BR IF LESS
 1452  003472  000413                          BR      M4              ;TO CHECK DRIVE
 1453  003474  023737  001226  001230  2$:     CMP     ENDTRK,BEGTRK   ;SEE IF ENDING TRACK EQ OR GT THAN BEGINNING
 1454  003502  103007                          BHIS    M4              ;BR IF YES
```

```
1455  003504  104401  022104          3$:     TYPE    ,MADRER         ;INVALID ADDRESS ENTERED
1456  003510  000755                          BR      M2              ;TRY AGAIN
1457  003512  012737  177777  001304  4$:     MOV     #-1,WRAP        ;SET WRAP AROUND FLAG
1458  003520  000400                          BR      M4
1459
1460                                  ;GO GET DATA PATTERN FOR FORMAT
1461
1462  003522  104401  022206          M4:     TYPE    ,MSELP          ;GO TYPE 'SELECT PATTERN'
1463  003526  104411                          RDLIN                   ;READ THE KEYBOARD
1464  003530  012601                          MOV     (SP)+,R1        ;ENTRY ADDRESS
1465  003532  004537  007074                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
1466  003536  003572                          3$                      ;ILLEGAL CHARACTER
1467  003540  003552                          1$                      ;CARRIAGE RETURN
1468  003542  003572                          3$                      ;" "
1469  003544  003552                          1$                      ;"'"
1470  003546  003564                          2$                      ;DIGIT 0-7
1471  003550  003572                          3$                      ;DIGIT 8-9
1472  003552  104401  022353          1$:     TYPE    ,MPATD          ;TYPE DEFAULT PATTERN
1473  003556  012702  000002                  MOV     #2,R2           ;WORST CASE PATTERN
1474  003562  000406                          BR      4$              ;GO SAVE PATTERN
1475  003564  020227  000002          2$:     CMP     R2,#2           ;IS # LARGER THAN 2
1476  003570  101403                          BLOS    4$              ;BRANCH IF NOT
1477  003572  104401  001166          3$:     TYPE    ,$QUES          ;TYPE '?'
1478  003576  000751                          BR      M4              ;RETYPE LINE
1479  003600  010237  001240          4$:     MOV     R2,PATSEL       ;SAVE PATTERN SELECTED
1480
1481                                  ;GO TYPE 'STARTING FORMAT ON DRIVE N'
1482
1483  003604  012737  006546  000060  M5:     MOV     #NEWSVC,@#TKVEC ;VECTOR FOR CONTROL-O
1484  003612  005037  177776                  CLR     @#PSW           ;LEVEL 0
1485  003616  005737  001220                  TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
1486  003622  001403                          BEQ     1$              ;BR IF 'CHECK' MODE
1487  003624  104401  022366                  TYPE    ,MSFOU          ;TYPE 'STARTING FORMAT ON DRIVE N'
1488  003630  000402                          BR      2$
1489  003632  104401  022423          1$:     TYPE    ,MSCHK          ;TYPE 'STARTING CHECK ON DRIVE N'
1490  003636                          2$:
1491  003636  013746  001214                  MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
1492                                                                  ;;TYPE DRIVE NUMBER
1493  003642  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1494  003644     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1495  003645     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1496  003646  104401  001167                  TYPE    ,$CRLF          ;CR-LF
1497
1498                                  ;SETUP TOTAL TRACK COUNT FOR FORMAT
1499
1500  003652  023737  001222  001224  CKADRS: CMP     ENDCYL,BEGCYL   ;STARTING AND ENDING CYLINDERS THE SAME ?
1501  003660  001011                          BNE     5$              ;BRANCH IF BEGCYL NOT EQUAL TO ENDCYL
1502  003662  013737  001226  001232          MOV     ENDTRK,TTRKS    ;END TRACK ADDRESS
1503  003670  163737  001230  001232          SUB     BEGTRK,TTRKS    ;SUBTRACT THE STARTING TRACK ADDRESS
1504  003676  005237  001232                  INC     TTRKS           ;MAKE THE NUMBER OF TRACKS INCLUSIVE
1505  003702  000440                          BR      SETPAT          ;SETUP THE DATA PATTERN
1506  003704  005737  001304          5$:     TST     WRAP            ;WRAP AROUND FLAG SET ?
1507  003710  001407                          BEQ     1$              ;NO
1508  003712  012702  001466                  MOV     #822.,R2        ;INITIAL COUNTER TO 822
1509  003716  163702  001224                  SUB     BEGCYL,R2       ;FIRST PART OF CYL #
1510  003722  063702  001222                  ADD     ENDCYL,R2       ;SECOND PART OF CYL #
```

```
1511  003726  000405                          BR      4$
1512  003730  013702  001222          1$:     MOV     ENDCYL,R2       ;ENDING CYLINDER
1513  003734  163702  001224                  SUB     BEGCYL,R2       ;SUBTRACT THE STARTING CYLINDER
1514  003740  005302                          DEC     R2              ;EXCLUSIVE LAST CYLINDER
1515  003742  012737  000005  001232  4$:     MOV     #5,TTRKS        ;INITATE COUNTER
1516  003750  163737  001230  001232          SUB     BEGTRK,TTRKS    ;# OF TRACK ON STARTING CYLINDER
1517  003756  063737  001226  001232          ADD     ENDTRK,TTRKS    ;# OF TRACK ON ENDING CYLINDER
1518  003764  005237  001232                  INC     TTRKS           ;INCLUSIVE ONE TRACK
1519  003770  005302                  3$:     DEC     R2              ;DECREMENT THE CYLINDER COUNT
1520  003772  100404                          BMI     SETPAT          ;BR IF DONE
1521  003774  062737  000005  001232          ADD     #5.,TTRKS       ;ADD CYLINDER WORTH OF TRACKS
1522  004002  000772                          BR      3$              ;CONTINUE
1523
1524                                  ;THIS CODE SETS UP THE SELECTED DATA PATTERN TO BE WRITTEN ON EACH SECTOR IMAGE
1525
1526  004004  005037  001242          SETPAT: CLR     PATA            ;CLEAR DATA PATTERN A
1527  004010  005037  001244                  CLR     PATB            ;CLEAR DATA PATTERN B
1528  004014  005737  001240                  TST     PATSEL          ;SEE IF PATTERN OF ONES
1529  004020  001420                          BEQ     1$              ;BR IF SO
1530  004022  012737  040502  001242          MOV     #040502,PATA    ;PATA = AB
1531  004030  012737  040644  001244          MOV     #040644,PATB    ;PATB=CD
1532  004036  022737  000001  001240          CMP     #1,PATSEL       ;SEE IF PATTERN OF ZEROS
1533  004044  001406                          BEQ     1$              ;BRANCH IF SO
1534  004046  012737  165555  001242          MOV     #165555,PATA    ;SET UP WORST CASE
1535  004054  012737  133333  001244          MOV     #133333,PATB    ;SET UP WORST CASE
1536  004062  013703  001256          1$:     MOV     WC,R3           ;SET UP COUNTER
1537  004066  012700  025732                  MOV     #BUFP,R0        ;SET UP MEMORY POINTER
1538  004072  013746  001242                  MOV     PATA,-(SP)      ;SAVE PATTERN A
1539  004076  013746  001244                  MOV     PATB,-(SP)      ;SAVE PATTERN B
1540  004102  032737  000001  001224          BIT     #BIT00,BEGCYL   ;A EVEN CYL # ?
1541  004110  001404                          BEQ     3$              ;YES
1542  004112  005137  001242                  COM     PATA            ;NO,COMPLEMENT PATTERN
1543  004116  005137  001244                  COM     PATB            ;COMPLEMENT PATTERN
1544  004122  013701  001242          3$:     MOV     PATA,R1         ;SET UP PATTERN IN R1
1545  004126  013702  001244                  MOV     PATB,R2         ;SET UP PATTERN IN R2
1546  004132  010120                  2$:     MOV     R1,(R0)+        ;MOV 1ST PAT INTO MEM
1547  004134  010220                          MOV     R2,(R0)+        ;MOV 2ND PAT INTO MEM
1548  004136  005303                          DEC     R3              ;KEEP COUNT
1549  004140  005303                          DEC     R3              ;KEEP COUNT
1550  004142  001373                          BNE     2$              ;DO IT AGAIN IF R3 NOT 0
1551  004144  012637  001244                  MOV     (SP)+,PATB      ;RESTORE PATTERN B
1552  004150  012637  001242                  MOV     (SP)+,PATA      ;RESTORE PATTERN A
1553
1554                                  ;THIS CODE SETS UP FOR THE ACTUAL FORMAT
1555
1556  004154  004737  005640          WRTRK:  JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
1557  004160  004737  006122                  JSR     PC,SETHDR       ;GO INITIALIZE HEADERS IN THE SECTOR BUFFER IN CORE
1558  004164  112737  000020  001357          MOVB    #20,FMTDPB+1    ;LOAD FMT22 BIT
1559  004172  005737  001264                  TST     SEC30           ;18 BIT MODE ?
1560  004176  001002                          BNE     1$              ;BR IF NOT
1561  004200  105037  001357                  CLRB    FMTDPB+1        ;CLEAR THE FMT22 BIT
1562  004204  112737  000143  001360  1$:     MOVB    #SETFMT,FMTDPB+2  ;'LOAD FORMAT' COMMAND
1563  004212  004037  013436          2$:     JSR     R0,RM01         ;START THE COMMAND
1564  004216  001356                          FMTDPB                  ;DPB ADDRESS
1565  004220  000774                          BR      2$              ;QUEUE FULL RETURN
1566  004222  005737  001374          3$:     TST     FMTDPB+16       ;LOOK FOR DONE
```

```
1567  004226  001775                          BEQ     3$              ;LOOP UNTIL FINISHED
1568  004230  005037  001216          WRTRK1: CLR     SOFSW           ;CLEAR ERROR COUNTER
1569  004234  005037  001250                  CLR     RETRY           ;ZERO THE RETRY COUNTER
1570  004240  105037  001366                  CLRB    FMTDPB+10       ;RESTORE SECTOR
1571  004244  013737  001260  001362          MOV     MWC,FMTDPB+4    ;RESTORE WC
1572  004252  012737  025732  001364          MOV     #BUFP,FMTDPB+6  ;RESTORE BA
1573  004260  005737  001220                  TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
1574  004264  001450                          BEQ     CKTRK           ;BR IF 'CHECK' MODE
1575  004266  112737  000163  001360  WRTRK2: MOVB    #WRTHD,FMTDPB+2 ;SET WRITE HEADER & DATA COMMAND IN TBL
1576  004274  012737  004274  001110          MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
1577  004302  012706  001100                  MOV     #STACK,SP       ;LOAD STACK POINTER
1578  004306  004037  013436          1$:     JSR     R0,RM01         ;GO FORMAT A TRACK
1579  004312  001356                          FMTDPB                  ;ADRS OF PARAMETERS - TBL
1580  004314  000774                          BR      1$              ;WAIT FOR QUEUE IF FULL
1581  004316  005737  001374          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
1582  004322  001775                          BEQ     2$              ;BRANCH IF NOT DONE
1583  004324  100024                          BPL     4$              ;BRANCH IF NO ERROR
1584  004326  012737  004354  001160          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
1585  004334  004737  005472                  JSR     PC,ERINDX       ;SEE WHICH ERROR
1586  004340  104010                          ERROR   10              ;DRIVE OFFLINE
1587  004342  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
1588  004344  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
1589  004346  104013                          ERROR   13              ;SOFTWARE TIMEOUT
1590  004350  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
1591  004352  104015                          ERROR   15              ;DRIVE/CONTROLLER ERROR DURING WRITE
1592  004354  004737  005572          3$:     JSR     PC,LOP.CK       ;LOOP ON THE ERROR ?
1593  004360  022737  000003  001250          CMP     #3,RETRY        ;ERROR RETRY LIMIT ?
1594  004366  001403                          BEQ     4$              ;BR IF REACHED
1595  004370  005237  001250                  INC     RETRY           ;COUNT THE ERROR
1596  004374  000744                          BR      1$              ;TRY AGAIN
1597  004376  005037  001160          4$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE ADDRESS
1598  004402  005037  001250                  CLR     RETRY           ;CLEAR THE RETRY COUNTER
1599
1600                                  ;CHECK THE TRACK JUST WRITTEN
1601
1602  004406  112737  000153  001360  CKTRK:  MOVB    #WCKHD,FMTDPB+2 ;SET WRITE CHECK HEADER & DATA COMMAND IN TBL
1603  004414  012737  004414  001110          MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
1604  004422  012706  001100                  MOV     #STACK,SP       ;LOAD STACK POINTER
1605  004426  004037  013436          1$:     JSR     R0,RM01         ;GO CHECK THE TRACK JUST FORMATTED
1606  004432  001356                          FMTDPB                  ;ADRS OF PARAMETERS - TBL
1607  004434  000774                          BR      1$              ;WAIT FOR QUEUE IF FULL
1608  004436  005737  001374          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
1609  004442  001775                          BEQ     2$              ;BRANCH IF NOT DONE
1610  004444  100112                          BPL     8$              ;BRANCH IF NO ERROR
1611  004446  113737  001314  001252          MOVB    RM.REG+RMDA,SAVSEC  ;GET THE SECTOR ADDRESS
1612  004454  001005                          BNE     3$              ;BRANCH IF NOT SECTOR 0
1613  004456  013737  001266  001252          MOV     MAXSEC,SAVSEC   ;RESTORE TO LAST SECTOR +1
1614  004464  005237  001252                  INC     SAVSEC          ;INCREMENT TO LAST SECTOR +1
1615  004470  005337  001252          3$:     DEC     SAVSEC          ;ADJUST SECTOR TO THE ONE THAT FAILED
1616  004474  012737  004740  001160          MOV     #10$,$ESCAPE    ;;ESCAPE TO 10$ ON ERROR
1617  004502  004737  005472                  JSR     PC,ERINDX       ;SEE WHICH ERROR
1618  004506  104010                          ERROR   10              ;DRIVE OFFLINE
1619  004510  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
1620  004512  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
1621  004514  104013                          ERROR   13              ;SOFTWARE TIMEOUT
1622  004516  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
```

```
1623  004520  032737  040000  001316          BIT     #WCE,RM.REG+RMCS2  ;WRITE CHECK ERROR ?
1624  004526  001005                          BNE     4$                 ;BR IF SET
1625  004530  033727  001322                  BIT     RM.REG+RMER1,(PC)+ ;CHECK FOR DATA ERRORS
1626  004534  13C620                          .WORD   DCK!OPI!DTE!HCRC!HCE!FER  ;DATA ERROR BITS
1627  004536  001001                          BNE     4$                 ;BR IF SET
1628  004540  104016                          ERROR   16                 ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1629  004542  005737  001220          4$:     TST     MODE               ;FORMAT OR CHECK MODE ?
1630  004546  001407                          BEQ     5$                 ;BR IF CHECK
1631  004550  005737  001216                  TST     SOFSW              ;RETRYING THE SECTOR ?
1632  004554  001404                          BEQ     5$                 ;BR IF NOT
1633  004556  012737  004762  001160          MOV     #11$,$ESCAPE       ;;ESCAPE TO 11$ ON ERROR
1634  004564  104017                          ERROR   17                 ;SECTOR NOT ACCEPTABLE
1635  004566  005037  001160          5$:     CLR     $ESCAPE            ;CLEAR THE ERROR ESCAPE ADDRESS
1636  004572  032737  040000  001320          BIT     #ERR,RM.REG+RMDS   ;DATA ERROR ?
1637  004600  001002                          BNE     6$                 ;BR IF IT IS
1638  004602  104024                          ERROR   24                 ;WRITE CHECK ERROR
1639  004604  000401                          BR      7$                 ;CONTINUE
1640  004606  104020                  6$:     ERROR   20                 ;DATA ERROR DURING WRITE CHECK
1641  004610  013701  001252          7$:     MOV     SAVSEC,R1          ;FAILING SECTOR ADDRESS
1642  004614  113737  001252  001366          MOVB    SAVSEC,FMTDPB+10   ;SECTOR ADDRESS
1643  004622  006301                          ASL     R1                 ;SETUP INDEX TO ADDRESS WORDS
1644  004624  016137  001500  001364          MOV     ADRTBL(R1),FMTDPB+6  ;BUFFER ADDRESS FOR FAILING SECTOR
1645  004632  013746  001260                  MOV     MWC,-(SP)          ;GET MAXIMUM WORD COUNT VALUE (2'S COMP,
1646  004636  066116  001600                  ADD     WCTBL(R1),(SP)     ;ADD WORD COUNT THROUGH FAILING SECTOR
1647  004642  012637  001254                  MOV     (SP)+,SAVWC        ;STORE WORD COUNT FOR REMAINDER OF TRACK
1648  004646  005737  001220                  TST     MODE               ;FORMAT OR CHECK MODE ?
1649  004652  001421                          BEQ     9$                 ;BR IF CHECK
1650  004654  012737  177376  001362          MOV     #-258.,FMTDPB+4    ;WORD COUNT FOR 1 SECTOR
1651  004662  005237  001216                  INC     SOFSW              ;INDICATE THAT A RETRY IS IN PROGRESS
1652  004666  000137  004266                  JMP     WRTRK2             ;REFORMAT ERROR SECTOR
1653  004672  005737  001216          8$:     TST     SOFSW              ;RETRY IN PROGRESS ?
1654  004676  001431                          BEQ     11$                ;BR IF NOT
1655  004700  022737  000002  001216          CMP     #2,SOFSW           ;SEE IF LAST RETRY
1656  004706  001403                          BEQ     9$                 ;BR IF IT IS
1657  004710  005237  001216                  INC     SOFSW.             ;INCREMENT RETRY COUNT
1658  004714  000644                          BR      1$                 ;READ AGAIN
1659  004716  123737  001266  001366  9$:     CMPB    MAXSEC,FMTDPB+10   ;SEE IF LAST SECTOR ON THE TRACK
1660  004724  001416                          BEQ     11$                ;BR IF IT IS
1661  004726  004737  006074                  JSR     PC,SCAWC           ;SETUP TO CHECK REMAINING SECTORS ON THE TRACK
1662  004732  005037  001216                  CLR     SOFSW              ;CLEAR RETRY COUNTER
1663  004736  000633                          BR      1$                 ;FINISH CHECKING THE TRACK
1664  004740  004737  005572          10$:    JSR     PC,LOP.CK          ;CHECK FOR LOOP ON ERROR
1665  004744  022737  000003  001250          CMP     #3,RETRY           ;ERROR RETRY REACHED ?
1666  004752  001403                          BEQ     11$                ;BR IF IT IS
1667  004754  005237  001250                  INC     RETRY              ;COUNT THE RETRY
1668  004760  000622                          BR      1$                 ;DO THE WRITE CHECK AGAIN
1669  004762  005037  001160          11$:    CLR     $ESCAPE            ;CLEAR THE ERROR RETURN ESCAPE ADDRESS
1670  004766  005037  001250                  CLR     RETRY              ;CLEAR THE RETRY COUNTER
1671  004772  032777  000002  174140          BIT     #BIT1,@SWR         ;LOOP ON CURRENT TRACK ?
1672  005000  001402                          BEQ     12$                ;BR IF NOT
1673  005002  000137  004230                  JMP     WRTRK1             ;START AGAIN ON THE SAME TRACK
1674  005006  004737  005730          12$:    JSR     PC,TRKTST          ;GET UPDATED TRACK AND CYLINDER ADDRESSES
1675  005012  000402                          BR      HDREAD             ;RETURN HERE IF DONE - DO QUICK CHECK OF DISK
1676  005014  000137  004230                  JMP     WRTRK1             ;CONTINUE WITH THE FORMAT
1677
1678                                  ;THIS CODE MAKES SURE EACH CYLINDER FORMATTED CONTAINS THE
```

```
1679                                    ;PROPER CYLINDER ADDRESS. THE PROGRAM IS LOOKING FOR
1680                                    ;POSSIBLE POSITIONER ERRORS THAT MAY HAVE OCCURRED DURING THE FORMAT.
1681
1682  005020  005037  001262      HDREAD: CLR     HEDERR          ;CLEAR HEADER CHECK ERROR INDICATOR
1683  005024  004737  005640              JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
1684  005030  004737  006122              JSR     PC,SETHDR       ;GO SET UP HEADERS IN CORE
1685  005034  013737  001224  001246      MOV     BEGCYL,CYLCK    ;USE 'BEGCYL' FOR FORMAT VERIFICATION
1686  005042  012737  177776  001362      MOV     #-2,FMTDPB+4    ;SET UP WORD COUNT
1687  005050  012737  025722  001364      MOV     #RBUF,FMTDPB+6  ;SET UP BUFFER ADRS
1688  005056  005037  001366              CLR     FMTDPB+10       ;CLEAR THE SECTOR & TRACK ADDRESS FIELD
1689  005062  013737  001224  001370      MOV     BEGCYL,FMTDPB+12  ;SETUP THE CYLINDER FIELD
1690  005070  112737  000173  001360      MOVB    #RDHD,FMTDPB+2  ;SET UP READ HEADER & DATA COMMAND
1691  005076  012737  005076  001110      MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
1692  005104  012706  001100              MOV     #STACK,SP       ;LOAD STACK POINTER
1693  005110  004037  013436      1$:     JSR     R0,RM01         ;GO READ HEADER
1694  005114  001356                      FMTDPB                  ;ADRS OF PARAMETER TBL
1695  005116  000774                      BR      1$              ;WAIT IF QUEUE IS FULL
1696  005120  005737  001374      2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
1697  005124  001775                      BEQ     2$              ;BRANCH IF NOT DONE
1698  005126  100024                      BPL     4$              ;BR IF NOT ERROR
1699  005130  012737  005212  001160      MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
1700  005136  004737  005472              JSR     PC,ERINDX       ;SEE WHICH ERROR
1701  005142  104010                      ERROR   10              ;DRIVE OFFLINE
1702  005144  104011                      ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
1703  005146  104012                      ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
1704  005150  104013                      ERROR   13              ;SOFTWARE TIMEOUT
1705  005152  104014                      ERROR   14              ;DRIVE UNSAFE ERROR
1706  005154  032737  177577  001374      BIT     #^C<HCE>,FMTDPB+16  ;IS ONLY ERROR A HEADER COMPARE ERROR ?
1707  005162  001401                      BEQ     3$              ;BR IF YES
1708  005164  104021                      ERROR   21              ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
1709  005166  005037  001160      3$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
1710  005172  104022                      ERROR   22              ;HEADER COMPARE ERROR VERIFYING HEADERS
1711  005174  004737  005572              JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
1712  005200  023737  025722  025732  4$: CMP     RBUF,BUFP       ;SEE IF CYL READ EQUALS CYL EXPECTED
1713  005206  001403                      BEQ     6$              ;BR IF CYL CORRECT
1714  005210  104023                      ERROR   23              ;CYLINDER NOT CORRECT
1715  005212  004737  005572      5$:     JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
1716  005216  023737  001222  001246  6$: CMP     ENDCYL,CYLCK    ;SEE IF LAST CYLINDER
1717  005224  001002                      BNE     7$              ;BR IF NOT FINISHED
1718  005226  000137  005336              JMP     $EOP            ;END OF FORMAT
1719  005232  005237  001246      7$:     INC     CYLCK           ;INCREMENT CYLINDER ADDRESS BEING CHECKED
1720  005236  022737  001467  001246      CMP     #823.,CYLCK     ;LAST CYLINDER ?
1721  005244  003027                      BGT     8$              ;NO
1722  005246  005037  001246              CLR     CYLCK           ;YES LAST CYLINDER
1723  005252  005037  001370              CLR     FMTDPB+12       ;RESER CYLINDER # TO 0
1724  005256  005037  001366              CLR     FMTDPB+10       ;RESET TRACK AND SECTOR #
1725  005262  005037  001374              CLR     FMTDPB+16       ;RESET CYLINDER #
1726  005266  013746  001224              MOV     BEGCYL,-(SP)  ;SAVE BEGCYL
1727  005272  013746  001230              MOV     BEGTRK,-(SP)    ;SAVE BEGTRK
1728  005276  005037  001230              CLR     BEGTRK
1729  005302  005037  001224              CLR     BEGCYL
1730  005306  004737  006122              JSR     PC,SETHDR       ;SET UP BUFFER
1731  005312  012637  001230              MOV     (SP)+,BEGTRK    ;RESTORE BEGTRK
1732  005316  012637  001224              MOV     (SP)+,BEGCYL    ;RESTORE BEGCYL
1733  005322  000672                      BR      1$
1734  005324  004737  006176      8$:     JSR     PC,UPDACY       ;SET UP FOR NEXT CYL
```

```
1735  005330  005237  001370                  INC     FMTDPB+12       ;ADVANCE TO NEXT CYL #
1736  005334  000665                          BR      1$              ;GO READ NEXT HEADER
1737
1738                                  .SBTTL  END OF PASS ROUTINE
1739
1740                                  ;;*****************************************************************
1741                                  ;*INCREMENT THE PASS NUMBER ($PASS)
1742                                  ;*IF THERES A MONITOR GO TO IT
1743                                  ;*IF THERE ISN'T JUMP TO DONE
1744
1745  005336                          $EOP:
1746  005336  005737  001220                  TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
1747  005342  001403                          BEQ     3$              ;BR IF 'CHECK'
1748  005344  104401  022457                  TYPE    .MFCMPT         ;'FORMAT COMPLETE'
1749  005350  000402                          BR      4$              ;FINISH THE END MESSAGE
1750  005352  104401  022504          3$:     TYPE    .MCCMPT         ;'CHECK COMPLETE'
1751  005356  104401  022530          4$:     TYPE    .NUMERR         ;'TOTAL ERRORS DETECTED: '
1752  005362  013746  001112                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
1753  005366  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
1754  005370  104401  001167                  TYPE    .$CRLF          ;CR-LF
1755  005374  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
1756  005400  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
1757  005406  005327                          DEC     (PC)+           ;;LOOP?
1758  005410  000001          $EOPCT: .WORD   1
1759  005412  003013                          BGT     $DOAGN          ;;YES
1760  005414  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
1761  005416  000001          $ENDCT: .WORD   1
1762  005420  005410                          $EOPCT
1763  005422  013700  0C0042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
1764  005426  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
1765  005430  000005                          RESET                   ;;CLEAR THE WORLD
1766  005432  004710          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
1767  005434  000240                          NOP                     ;;SAVE ROOM
1768  005436  000240                          NOP                     ;;FOR
1769  005440  000240                          NOP                     ;;ACT11
1770  005442                          $DOAGN:
1771  005442  000137                          JMP     @(PC)+          ;;RETURN
1772  005444  005446          $RTNAD: .WORD   DONE
1773  005446  032777  000001  173464  DONE:   BIT     #SW0,@SWR       ;SEE IF SWR 0 SET
1774  005454  001002                          BNE     1$              ;BR IF SET
1775  005456  000137  003174                  JMP     M0              ;ASK FOR NEXT DRIVE
1776  005462  012706  001100          1$:     MOV     #STACK,SP       ;RESET STACK POINTER
1777  005466  000137  003604                  JMP     M5              ;DO THE FORMAT OR CHECK AGAIN
1778
1779                                  ;;*****************************************************************
1780
1781                                  .SBTTL  SUPPORT SUBROUTINES
1782
1783                                  ;;*****************************************************************
1784
1785                                  ;THIS ROUTINE DETERMINES THE ERROR TYPE
1786
1787  005472  010146          ERINDX: MOV     R1,-(SP)        ;STORE R1
1788  005474  005001                  CLR     R1              ;CLEAR R1
1789  005476  033727  001374          BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
1790  005502  060036                  .WORD   BIT14!BIT13!BIT2!BIT1  ;DRIVE OFFLINE
```

```
1791  005504  001025                        BNE     5$                ;BR IF OFFLINE
1792  005506  033727  001374                BIT     FMTDPB+16,(PC)+   ;CHECK ERROR TYPE
1793  005512  010000                        .WORD   BIT12             ;DRIVE PERSISTENTLY UNSAFE
1794  005514  001020                        BNE     4$                ;BR IF UNSAFE
1795  005516  033727  001374                BIT     FMTDPB+16,(PC)+   ;CHECK ERROR TYPE
1796  005522  006000                        .WORD   BIT11!BIT10       ;UNCORRECTABLE MASSBUS PARITY ERROR ?
1797  005524  001013                        BNE     3$                ;BR IF PARITY ERROR
1798  005526  033727  001374                BIT     FMTDPB+16,(PC)+   ;CHECK ERROR TYPE
1799  005532  001400                        .WORD   BIT09!BIT08       ;SOFTWARE TIMEOUT ?
1800  005534  001006                        BNE     2$                ;BR IF YES
1801  005536  033727  001374                BIT     FMTDPB+16,(PC)+   ;CHECK ERROR TYPE
1802  005542  000020                        .WORD   BIT04             ;DRIVE UNSAFE ERROR ?
1803  005544  001001                        BNE     1$                ;BR IF YES
1804  005546  005201                        INC     R1                ;INCREMENT THE RETURN INDEX
1805  005550  005201              1$:       INC     R1                ;INCREMENT THE RETURN INDEX
1806  005552  005201              2$:       INC     R1                ;INCREMENT THE RETURN INDEX
1807  005554  005201              3$:       INC     R1                ;INCREMENT THE RETURN INDEX
1808  005556  005201              4$:       INC     R1                ;INCREMENT THE RETURN INDEX
1809  005560  006301              5$:       ASL     R1                ;DOUBLE THE INCREMENT
1810  00[illegible]  060166  000002          ADD     R1,2(SP)          ;DEVELOP THE RETURN ADDRESS
1811  00[illegible]  012601                  MOV     (SP)+,R1          ;RESTORE R1
1812  005570  000207                        RTS     PC                ;RETURN
1813
1814                                ;ROUTINE TO CHECK FOR LOOP ON ERROR
1815
1816  005572  032777  001000  173340  LOP.CK: BIT     #SWD9,@SWR        ;LOOP ON ERROR ?
1817  005600  001402                        BEQ     1$                ;BR IF NOT
1818  005602  000177  173302                JMP     @$LPERR           ;GO TO THE LOOP ON ERROR ADDRESS
1819  005606  005037  001160              1$:       CLR     $ESCAPE           ;CLEAR THE ERROR ESCAPE ADDRESS
1820  005612  033727  001374                BIT     FMTDPB+16,(PC)+   ;CHECK FOR 'FATAL' ERROR
1821  005616  072002                        .WORD   BIT14!BIT13!BIT12!BIT10!BIT01  ;'FATAL' ERROR BITS
1822  005620  001004                        BNE     2$                ;BR IF NOT
1823  005622  032737  004000  001322        BIT     #WLE.RM.REG+RMER,@SWR  ;WRITE LOCK ERROR ?
1824  005630  001402                        BEQ     3$                ;BR IF NOT
1825  005632  000137  005336              2$:       JMP     $EOP              ;TERMINATE THE FORMAT
1826  005636  000207                      3$:       RTS     PC                ;RETURN
1827
1828                                ;THIS ROUTINE SETS UP THE INPUT TABLE FOR THE DRIVER ROUTINE
1829
1830  005640  113737  001214  001356  SETTBL: MOVB    DRIVE,FMTDPB      ;SET UP DRIVE #
1831  005646  105037  001361                CLRB    FMTDPB+3          ;CLEAR HIGH ORDER ADRS BITS
1832  005652  013737  001260  001362        MOV     MWC,FMTDPB+4      ;LOAD UP WORD COUNT
1833  005660  012737  025732  001364        MOV     #BUFP,FMTDPB+6    ;LOAD UP CURRENT ADRS
1834  005666  105037  001366                CLRB    FMTDPB+10         ;SET SECTOR TO ZERO
1835  005672  113737  001230  001367        MOVB    BEGTRK,FMTDPB+11  ;SET UP STARTING TRK ADRS
1836  005700  013737  001224  001370        MOV     BEGCYL,FMTDPB+12  ;SET UP STARTING CYL
1837  005706  005037  001374                CLR     FMTDPB+16         ;CLEAR RM01 STATUS
1838  005712  013737  001230  001236        MOV     BEGTRK,TRKCNT     ;SET UP PARTIAL CYL TRACK COUNT
1839  005720  013737  001232  001234        MOV     TTRKS,TTRKSC      ;SET UP TOTAL TRACKS COUNTER
1840  005726  000207                        RTS     PC                ;RETURN FROM SETUP
1841
1842                                ;THIS ROUTINE CONTROLS THE DISK ADDRESSING AND TOTAL TRK COUNT
1843                                ;IT IS ENTERED AFTER EVERY TRK OPERATION
1844
1845  005730  005337  001234          TRKTST: DEC     TTRKSC            ;SEE IF LAST TRACK
1846  005734  001456                        BEQ     3$                ;BRANCH IF LAST TRACK
```

```
1847  005736  022737  000004  001236          CMP     #4.,TRKCNT      ;IS THIS THE LAST TRACK IN THE CYLINDER ?
1848  005744  001407                          BEQ     1$              ;BRANCH IF SO
1849  005746  005237  001236                  INC     TRKCNT          ;COUNT UP TRACK WITHIN CYLINDER
1850  005752  105237  001367                  INCB    FMTDPB+11       ;INCREMENT TRACK NUMBER
1851  005756  004737  006322                  JSR     PC,UPDATK       ;UPDATE TRACK ADDRESS IN BUFFER
1852  005762  000441                          BR      2$              ;EXIT
1853  005764  005037  001236          1$:     CLR     TRKCNT          ;CLEAR TRACK COUNT FOR NEXT CYLINDER
1854  005770  105037  001367                  CLRB    FMTDPB+11       ;RESET TRACK ADDRESS TO 0
1855  005774  005237  001370                  INC     FMTDPB+12       ;UPDATE CYLINDER NUMBER
1856  006000  022737  001467  001370          CMP     #823.,FMTDPB+12 ;LAST CYLINDER ?
1857  006006  003025                          BGT     4$              ;NO
1858  006010  013746  001224                  MOV     BEGCYL,-(SP)    ;SAVE BEGCYL
1859  006014  013746  001230                  MOV     BEGTRK,-(SP)    ;SAVE BEGTRK
1860  006020  005037  001224                  CLR     BEGCYL          ;RESET THE STARTING CYL
1861  006024  005037  001230                  CLR     BEGTRK          ;RESET THE START TRACK
1862  006030  004737  006122                  JSR     PC,SETHDR       ;INITIATE HEAD WORDS
1863  006034  005037  001366                  CLR     FMTDPB+10       ;CLEAR TRACK AND SECTOR
1864  006040  005037  001370                  CLR     FMTDPB+12       ;CLEAR CYLINDER #
1865  006044  005037  001374                  CLR     FMTDPB+16       ;CLEAR STATUS
1866  006050  012637  001230                  MOV     (SP)+,BEGTRK    ;RESTORE BEGTRK
1867  006054  012637  001224                  MOV     (SP)+,BEGCYL    ;RESTORE BEGCYL
1868  006060  000402                          BR      2$
1869  006062  004737  006176          4$:     JSR     PC,UPDACY       ;UPDATE CYLINDER NUMBER IN BUFFER
1870  006066  062716  000002          2$:     ADD     #2,(SP)         ;ADD TWO TO RETURN ADRS
1871  006072  000207                  3$:     RTS     PC              ;RETURN
1872
1873                                  ;THIS ROUTINE SETS UP WC & BA FOR REMAINING SECTORS
1874                                  ;AFTER A WRITE CHECK ERROR
1875
1876  006074  013737  001254  001362  SCAWC:  MOV     SAVWC,FMTDPB+4  ;SET UP WC FOR REMAINING SECTORS
1877  006102  062737  001004  001364          ADD     #516.,FMTDPB+6  ;SET UP BA FOR REMAINING SECTORS
1878  006110  005237  001366                  INC     FMTDPB+10       ;ADVANCE TO NEXT SECTOR IN TBL
1879  006114  005037  001216                  CLR     SOFSW           ;RESET RETRY COUNTER
1880  006120  000207                          RTS     PC              ;RETURN TO COMPLETE TRK FORMAT
1881
1882
1883                                  ;THIS ROUTINE SETS UP THE CYLINDER ADRS, FORMAT BIT, TRACK AND
1884                                  ;SECTOR ADRS IN MEMORY WITH THE STARTING CYL - TRK INFORMATION
1885
1886  006122  013701  001266          SETHDR: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
1887  006126  012700  025732                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
1888  006132  013702  001224                  MOV     BEGCYL,R2       ;PUT STARTING CYL # IN R2
1889  006136  013703  001230                  MOV     BEGTRK,R3       ;PUT STARTING TRK # IN R3
1890  006142  000303                          SWAB    R3              ;JUSTIFY TRACK ADRS
1891  006144  005737  001264                  TST     SEC30           ;SEE IF 30 OR 32 SECTOR MODE
1892  006150  001402                          BEQ     1$              ;BR IF 30 SECTOR MODE
1893  006152  052702  010000                  BIS     #10000,R2       ;SET THE 32 SECTOR FORMAT BIT
1894  006156  010220          1$:     MOV     R2,(R0)+        ;WRITE IN HEADER AREA OF CORE THE CYL ADRS
1895  006160  010320                  MOV     R3,(R0)+        ;WRITE IN HEADER AREA OF CORE THE TRK ADRS
1896  006162  062700  001000                  ADD     #512.,R0        ;SET UP FOR NEXT HEADER
1897  006166  005203                          INC     R3              ;UPDATE SECTOR ADRS FOR NEXT HEADER
1898  006170  005301                          DEC     R1              ;MAINTAIN COUNT OF SECTORS
1899  006172  002371                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
1900  006174  000207                          RTS     PC              ;EXIT - HEADERS ARE LOADED INTO CORE
1901
1902                                  ;THIS ROUTINE UPDATES THE CYLINDER ADRS OF THE HEADER WORDS IN CORE
```

```
1903
1904  006176  013701  001266          UPDACY: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
1905  006202  012700  025732                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
1906  006206  010246                          MOV     R2,-(SP)        ;SAVE R2
1907  006210  013746  001242                  MOV     PATA,-(SP)      ;SAVE PATTERN A
1908  006214  013746  001244                  MOV     PATB,-(SP)      ;SAVE PATTERN B
1909  006220  005220                  1$:     INC     (R0)+           ;INCREMENT FOR NEXT CYLINDER
1910  006222  042720  177400                  BIC     #177400,(R0)+   ;RESET TRK ADRS TO 0
1911  006226  012702  001000                  MOV     #512.,R2        ;DATA FIELD LENGTH
1912  006232  011637  001244                  MOV     (SP),PATB       ;SET PATTERN
1913  006236  016637  000002  001242          MOV     2(SP),PATA      ;SET PATTERN
1914  006244  032760  000001  177774          BIT     #BIT00,-4(R0)   ;EVEN CYLINDER # ?
1915  006252  001404                          BEQ     2$              ;YES,USE ORIGINAL PATTERN
1916  006254  005137  001242                  COM     PATA            ;ODD CYLINDER #,USE COMPLEMENT
1917  006260  005137  001244                  COM     PATB            ;DATA PATTERN
1918  006264  013720  001242          2$:     MOV     PATA,(R0)+      ;FILL BUFFER
1919  006270  013720  001244                  MOV     PATB,(R0)+      ;FILL BUFFER
1920  006274  162702  000004                  SUB     #4,R2           ;END OF DATA FIELD ?
1921  006300  001371                          BNE     2$              ;NO
1922  006302  005301                          DEC     R1              ;COUNT SECTORS
1923  006304  002345                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
1924  006306  012637  001244                  MOV     (SP)+,PATB      ;RESTORE PATB
1925  006312  012637  001242                  MOV     (SP)+,PATA      ;RESTORE PATA
1926  006316  012602                          MOV     (SP)+,R2        ;RESETORE THE R2
1927  006320  000207                          RTS     PC              ;EXIT
1928
1929                                  ;THIS ROUTINE UPDATES THE TRACK ADRS OF THE HEADER WORDS IN CORE
1930
1931  006322  013701  001266          UPDATK: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
1932  006326  012700  025732                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
1933  006332  005720                          TST     (R0)+           ;POINT HEADER POINTER TO TRK - SEC ADRS
1934  006334  062710  000400          1$:     ADD     #400,(R0)       ;INDEX TRK ADRS
1935  006340  062700  001004                  ADD     #516.,R0        ;SET UP FOR NEXT HEADER
1936  006344  005301                          DEC     R1              ;COUNT SECTORS
1937  006346  002372                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
1938  006350  000207                          RTS     PC              ;EXIT
1939
1940                                  ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
1941
1942  006352  012737  006430  000004  ST.CLK: MOV     #STCLK1,@#ERRVEC  ;SET UP VECTOR FOR P CLK
1943  006360  005037  000006                  CLR     @#ERRVEC+2      ;NEW PSW
1944  006364  005777  172606                  TST     @$LKCSR         ;CHECK FOR KW11-P
1945  006370  013746  001202                  MOV     $LPVEC,-(SP)    ;VECTOR ADDRESS
1946  006374  012776  006514  000000          MOV     #CLOCK,@(SP)    ;SET UP KW11-P VECTOR
1947  006402  062716  000002                  ADD     #2,(SP)         ;POINT TO PSW
1948  006406  012736  000300                  MOV     #PR6,@(SP)+     ;PSW - PRI 6
1949  006412  012777  177777  172560          MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER
1950  006420  012777  000135  172550          MOV     #135,@$LKCSR    ;SET CLK - CNT UP
1951  006426  000426                          BR      STCLK3          ;
1952  006430  062706  000004          STCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
1953  006434  012737  006500  000004          MOV     #STCLK2,@#ERRVEC        ;CHANGE ERROR VECTOR
1954  006442  005777  172540                  TST     @$LKS           ;LOOK FOR KW11-L
1955  006446  013746  001210                  MOV     $LLVEC,-(SP)    ;KW11-L VECTOR ADDRESS
1956  006452  012776  006514  000000          MOV     #CLOCK,@(SP)    ;SET UP KW11-L VECTOR
1957  006460  062716  000002                  ADD     #2,(SP)         ;INCREMENT VECTOR ADDRESS
1958  006464  012736  000300                  MOV     #PR6,@(SP)+     ;PSW - PRI 6
```

```
1959  006470  012777  000100  172510          MOV     #100,@$LKS      ;SET KW11-L INTERRUPT ENABLE
1960  006476  000402                          BR      STCLK3
1961  006500  062706  000004          STCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
1962  006504  012737  000006  000004  STCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
1963  006512  000207                          RTS     PC
1964
1965                                  ;THIS CODE SERVICES A CLOCK INTERRUPT EVERY 16MS
1966
1967  006514  012746  000020          CLOCK:  MOV     #16.,-(SP)      ;PUT MILLISECONDS ON THE STACK
1968  006520  004737  017124                  JSR     PC,RPTMR        ;GO REPORT TIME
1969  006524  000002                          RTI                     ;RETURN AND CONTINUE
1970
1971                                  ;ROUTINE TO INTERCEPT 'CONTROL C' TYPED DURING PARAMETER ENTRY TIME
1972
1973  006526  012706  001100          OENTER: MOV     #STACK,SP       ;INITIALIZE THE STACK
1974  006532  005737  012626          1$:     TST     TRNSWT          ;ALL ACTIVITY STOPPED ?
1975  006536  001375                          BNE     1$              ;BR IF NOT
1976  006540  000005                          RESET                   ;CLEAR THE BUS
1977  006542  000137  003134                  JMP     M1B             ;START AGAIN WITH DRIVE SELECTION
1978
1979                                  ;ROUTINE TO TYPE THE PRESENT DISK ADDRESS.  ENTERED BY TYPING 'CONTROL C'
1980
1981  006546  117746  172374          NEWSVC: MOVB    @$TKB,-(SP) ;READ FROM TTY BUFFER
1982  006552  042716  177600                  BIC     #^C177,(SP)     ;STRIP PARITY
1983  006556  022627  000017                  CMP     (SP)+,#15.      ;CONTROL-O ?
1984  006562  001406                          BEQ     1$              ;YES
1985  006564  005777  172356                  TST     @$TKB           ;CLEAR DONE BIT
1986  006570  012777  000100  172346          MOV     #100,@$TKS      ;ENABLE TTY INTERRUPT
1987  006576  000002                          RTI
1988  006600  004737  010652          1$:     JSR     PC,$TKINT       ;INITIAL VECTOR
1989  006604  005037  177776          TYPADR: CLR     PSW             ;SET PROCESSOR TO PRIORITY 0
1990  006610  012737  006526  001300          MOV     #OENTER,CNTLC   ;CHANGE 'CONTROL C' RETURN ADDRESS
1991  006616  104401  022560                  TYPE    .ADDRIS         ;'PRESENT ADDRESS IS: '
1992  006622  104401  021327                  TYPE    ,C              ;'C'
1993  006626  013746  001370                  MOV     FMTDPB+12.,-(SP)  ;PUT THE CYLINDER ADDRESS ON THE STACK
1994  006632  004737  012136                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
1995  006636  004737  012076                  JSR     PC,$SUPRS       ;TYPE IT
1996  006642  104401  021337                  TYPE    ,LINSP          ;SPACES
1997  006646  104401  021331                  TYPE    ,T              ;'T'
1998  006652  005046                          CLR     -(SP)           ;CLEAR THE STACK
1999  006654  113716  001367                  MOVB    FMTDPB+11.,(SP) ;PUT THE TRACK ADDRESS ON THE STACK
2000  006660  004737  012136                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
2001  006664  004737  012076                  JSR     PC,$SUPRS       ;TYPE IT
2002  006670  104401  001167                  TYPE    ,$CRLF          ;CR-LF
2003  006674  012737  006546  000060          MOV     #NEWSVC,@#TKVEC ;RESTORE ENTRANCE TO THIS ROUTINE
2004  006702  000002                          RTI                     ;RETURN
2005
2006                                  ;PARAMETER ENTRY ROUTINE
2007                                  ;CALL
2008                                  ;       MOV     #ADR,R3         ;PARAMETER TABLE ADDRESS
2009                                  ;       JSR     PC,PARENT       ;GET THE PARAMETERS
2010
2011  006704  010346          PARENT: MOV     R3,-(SP)        ;SAVE R3
2012  006706  012703  001376          MOV     #TABLE,R3       ;PARAMETER TABLE ADDRESS
2013  006712  012337  006722  1$:     MOV     (R3)+,3$        ;ADDRESS OF PARAMETER NAME
2014  006716  001436                  BEQ     9$              ;BR IF AT END OF TABLE
```

```
                                                TYPE                    ;TYPE THE PARAMETER NAME
                                        3$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
                                                MOV     (R3)+,R2        ;MAXIMUM PARAMETER VALUE
                                                MOV     (R3)+,R5        ;ADDRESS OF PARAMETER
                                                MOV     (R5),-(SP)      ;CURRENT VALUE OF PARAMETER
                                                TYPOS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
                                                TYPE    ,SLASH          ;' / '
                                                RDLIN                   ;READ THE KEYBOARD
                                                MOV     (SP)+,R1        ;INPUT ASCII STRING ADDRESS
                                                JSR     R5,CK.DIG       ;CHECK THE DIGIT(S)
                                                1$                      ;CARRIAGE RETURN ONLY ENTERED
                                                9$                      ;PERIOD ONLY ENTERED
                                                6$                      ;ILLEGAL INPUT
                                                5$                      ;TERMINATED WITH A CARRIAGE RETURN
                                                6$                      ;TERMINATED WITH A "."
                                                7$                      ;TERMINATED WITH A ","
                                        5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
                                                BR      1$              ;GET MORE PARAMETERS
                                        6$:     TYPE    ,BADENT         ;'BAD ENTRY'
                                                SUB     #6,R3           ;DECREMENT THE TABLE POINTER
                                                BR      1$              ;TRY AGAIN
                                        7$:     MOV     R2,(R5)         ;NEW VALUE
                                                BR      9$              ;EXIT
                                        8$:     MOV     #TABLE,R3       ;RELOAD THE PARAMETER TABLE ADDRESS
                                                BR      1$              ;TRY AGAIN
                                        9$:     MOV     (SP)+,R3        ;RESTORE R3
                                                RTS     PC              ;RETURN


                                        ;THIS ROUTINE IS USED TO CHECK IF AN
                                        ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R5,CK.OCT       ;CHECK THE CHARACTER
                                        ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
                                        ;       RETURN2                 ;CHARACTER IS IN R2 AS A
                                        ;                               ;OCTAL DIGIT

007020  121127  000060          CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
007024  103407                          BLO     1$              ;YES -- BRANCH
007026  121127  000067                  CMPB    (R1),#'7        ;GREATER THAN SEVEN?
007032  101004                          BHI     1$              ;YES -- BRANCH
007034  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
007036  042702  177770                  BIC     #^C7,R2         ;STRIP AWAY THE ASCII
007042  005725                          TST     (R5)+           ;ADJUST FOR RETURN
007044  000205                  1$:     RTS     R5              ;RETURN


                                        ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
                                        ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R5,CK.DEC       ;CHECK THE CHARACTER
                                        ;       RETURN1                 ;NOT BETWEEN 0 AND 9
                                        ;       RETURN2                 ;BETWEEN 0 AND 9
                                        ;                               ;R2 = DIGIT
```

```
2071  007046  121127  000060          CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
2072  007052  103407                          BLO     1$              ;YES -- BRANCH
2073  007054  121127  000071                  CMPB    (R1),#'9        ;GREATER THAN NINE?
2074  007060  101004                          BHI     1$              ;YES -- BRANCH
2075  007062  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
2076  007064  042702  000060                  BIC     #'0,R2          ;STRIP AWAY THE ASCII
2077  007070  005725                          TST     (R5)+           ;ADJUST FOR RETURN
2078  007072  000205                  1$:     RTS     R5              ;RETURN
2079
2080                                          ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
2081                                          ;DETERMINE WHAT IT IS.
2082                                          ;CALL
2083                                          ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
2084                                          ;       JSR     R5,CK.CHR       ;CHECK CHARACTER
2085                                          ;       RETURN  ADR1            ;UNKNOWN CHARACTER
2086                                          ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
2087                                          ;       RETURN  ADR3            ;COMMA * (R1)=ADR+1
2088                                          ;       RETURN  ADR4            ;PERIOD * (R1)=ADR+1
2089                                          ;       RETURN  ADR5            ;DIGIT BETWEEN 0 AND 7.
2090                                          ;       RETURN  ADR6            ;DIGIT BETWEEN 8 AND 9.
2091                                          ;                               ;R2 = DIGIT * (R1)=ADR+1
2092
2093  007074  105711                  CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
2094  007076  001417                          BEQ     3$              ;YES -- BRANCH
2095  007100  121127  000054                  CMPB    (R1),#',        ;"COMMA"?
2096  007104  001413                          BEQ     2$              ;YES -- BRANCH
2097  007106  121127  000056                  CMPB    (R1),#'.        ;"PERIOD"?
2098  007112  001407                          BEQ     1$              ;YES -- BRANCH
2099  007114  004537  007046                  JSR     R5,CK.DEC       ;"DIGIT"?
2100  007120  000410                          BR      4$              ;NO -- BRANCH
2101  007122  004537  007020                  JSR     R5,CK.OCT       ;OCTAL ?
2102  007126  005725                          TST     (R5)+           ;DIGIT BETWEEN 8-9
2103  007130  005725                          TST     (R5)+           ;DIGIT BETWEEN 0-7
2104  007132  005725                  1$:     TST     (R5)+           ;PERIOD
2105  007134  005725                  2$:     TST     (R5)+           ;COMMA
2106  007136  005725                  3$:     TST     (R5)+           ;CARRIAGE RETURN
2107  007140  005201                          INC     R1              ;MOVE POINTER TO NEXT CHARACTER
2108  007142  011505                  4$:     MOV     (R5),R5         ;UNKNOWN CHARACTER
2109  007144  000205                          RTS     R5              ;RETURN
2110
2111                                          ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
2112                                          ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
2113                                          ;CALL
2114                                          ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
2115                                          ;       MOV     #NUM,R2         ;MAX. MAGNITUDE OF INPUT NUMBER
2116                                          ;       JSR     R5,CK.DIG       ;CHECK DIGITS
2117                                          ;       RETURN  ADR1            ;"CR" ONLY ENTERED -- R2=0
2118                                          ;       RETURN  ADR2            ;"PERIOD" ONLY ENTERED -- R2=0
2119                                          ;       RETURN  ADR3            ;ILLEGAL CHARACTER OR INPUT TOO LARGE -- R2=?
2120                                          ;       RETURN  ADR4            ;"CR" -- R2 = NUMBER
2121                                          ;       RETURN  ADR5            ;"COMMA" -- R2 = NUMBER
2122                                          ;       RETURN  ADR6            ;"PERIOD" -- R2 = NUMBER
2123
2124  007146  010446                  CK.DIG: MOV     R4,-(SP)        ;SAVE R4
2125  007150  010346                          MOV     R3,-(SP)        ;SAVE R3
2126  007152  010246                          MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
```

```
2127  007154  005002                        CLR     R2              ;START WITH 0
2128  007156  005003                        CLR     R3
2129  007160  005004                        CLR     R4
2130  007162  004537  007074                JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
2131  007166  007262                        6$                      ;ILLEGAL CHARACTER
2132  007170  007270                        9$                      ;CARRIAGE RETURN
2133  007172  007262                        6$                      ;" "
2134  007174  007264                        7$                      ;"."
2135  007176  007202                        1$                      ;DIGIT 0-7
2136  007200  007202                        1$                      ;DIGIT 8-9
2137  007202  062705  000004        1$:     ADD     #4,R5           ;STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
2138  007206  006303        2$:     ASL     R3                      ;INPUT NUMBER *2
2139  007210  010346                MOV     R3,-(SP)                ;SAVE *2
2140  007212  006303                ASL     R3                      ;*4
2141  007214  006303                ASL     R3                      ;*8
2142  007216  062603                ADD     (SP)+,R3                ;(*2)+(*8) = *10
2143  007220  060203                ADD     R2,R3                   ;UPDATE THE INPUT NUMBER
2144  007222  004537  007074        JSR     R5,CK.CHR               ;CHECK ONE CHARACTER
2145  007226  007266                8$                              ;ILLEGAL CHARACTER
2146  007230  007252                5$                              ;CARRIAGE RETURN
2147  007232  007250                4$                              ;" "
2148  007234  007242                3$                              ;"."
2149  007236  007206                2$                              ;DIGIT 0-7
2150  007240  007206                2$                              ;DIGIT 8-9
2151  007242  105711        3$:     TSTB    (R1)                    ;DOES A "CR" FOLLOW THE "PERIOD"
2152  007244  001010                BNE     8$                      ;BR IF NOT
2153  007246  005724                TST     (R4)+                   ;INCREMENT THE RETURN
2154  007250  005724        4$:     TST     (R4)+                   ;INCREMENT THE RETURN
2155  007252  005724        5$:     TST     (R4)+                   ;INCREMENT THE RETURN
2156  007254  020316                CMP     R3,(SP)                 ;CHECK THE MAGNITUDE OF THE NUMBER
2157  007256  002004                BGE     9$                      ;BR IF ENTERED NUMBER TOO LARGE
2158  007260  000402                BR      8$                      ;BYPASS INCREMENT
2159  007262  005725        6$:     TST     (R5)+                   ;INCREMENT RETURN PAST INVALID RETURN
2160  007264  005725        7$:     TST     (R5)+                   ;INCREMENT RETURN
2161  007266  060405        8$:     ADD     R4,R5                   ;SETUP RETURN POINTER
2162  007270  010302        9$:     MOV     R3,R2                   ;ENTERED VALUE
2163  007272  005726                TST     (SP)+                   ;CLEAN MAX. SIZE OFF OF STACK
2164  007274  012603                MOV     (SP)+,R3                ;RESTORE R3
2165  007276  012604                MOV     (SP)+,R4                ;RESTORE R4
2166  007300  011505                MOV     (R5),R5                 ;GET RETURN ADDRESS
2167  007302  000205                RTS     R5                      ;RETURN
2168
2169                          .SBTTL  MACRO ROUTINES
2170
2171
2172                          .SBTTL  ERROR HANDLER ROUTINE
2173
2174                          ;;*****************************************************************
2175                          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
2176                          ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2177                          ;*AND GO TO TYPERR ON ERROR
2178                          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2179                          ;*SW15=1          HALT ON ERROR
2180                          ;*SW13=1          INHIBIT ERROR TYPEOUTS
2181                          ;*SW10=1          BELL ON ERROR
2182                          ;*SW03=1          LOOP ON ERROR
```

```
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
                                        ;
007304                                  $ERROR:
007304  104407                                 CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
007306  010137  001274                         MOV     R1,DDRIVE       ;DRIVE ADDRESS IF DRIVE ERROR CALL
007312  010337  001276                         MOV     R3,ATTN         ;ATTENTION REGISTER CONTENTS
007316  013737  001370  001270                 MOV     FMTDPB+12,DS.CYL  ;CURRENT CYLINDER ADDRESS
007324  113737  001367  001272                 MOVB    FMTDPB+11,DS.TRK  ;REQUESTED TRACK ADDRESS
007332  005737  001312                         TST     RM.REG+RMBA     ;NON-ZERO BUFFER ADDRESS ?
007336  001406                                 BEQ     7$              ;BR IF NO BUFFER ADDRESS
007340  013746  001312                         MOV     RM.REG+RMBA,-(SP)  ;BUFFER ADDRESS
007344  162716  000002                         SUB     #2,(SP)         ;DECREMENT THE ADDRESS
007350  013637  001124                         MOV     @(SP)+,$GDDAT   ;GET THE BUFFER WORD WHICH DIDN'T COMPARE
007354  105237  001103                 7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
007360  001775                                 BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
007362  013777  001102  171552                 MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
007370  032777  002000  171542                 BIT     #BIT10,@SWR     ;;BELL ON ERROR?
007376  001402                                 BEQ     1$              ;;NO - SKIP
007400  104401  001162                         TYPE    ,$BELL          ;;RING BELL
007404  005237  001112                 1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
007410  011637  001116                         MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
007414  162737  000002  001116                 SUB     #2,$ERRPC
007422  117737  171470  001114                 MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
007430  032777  020000  171502                 BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
007436  001004                                 BNE     20$             ;;SKIP TYPEOUTS
007440  004737  007512                         JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
007444  104401  001167                         TYPE    ,$CRLF
007450                                  20$:
007450  005777  171464                 2$:     TST     @SWR            ;;HALT ON ERROR
007454  100002                                 BPL     3$              ;;SKIP IF CONTINUE
007456  000000                                 HALT                    ;;HALT ON ERROR!
007460  104407                                 CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
007462  032777  001000  171450         3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
007470  001402                                 BEQ     4$              ;;BR IF NO
007472  013716  001110                         MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
007476  005737  001160                 4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
007502  001402                                 BEQ     5$              ;;BR IF NONE
007504  013716  001160                         MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
007510                                  5$:
007510  000002                                 RTI                     ;;RETURN
                                        ;
                                        ;
                                        ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
                                        ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
                                        ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
                                        ;CONCERNING THE ERROR.
                                        ;
007512  104412                          TYPERR: SAVREG                  ;SAVE R0-R5
007514  005000                                  CLR     R0              ;CLEAR R0 FOR ERROR NUMBER
007516  113700  001114                         MOVB    $ITEMB,R0       ;ERROR NUMBER
007522  005300                                 DEC     R0              ;FORM INDEX FOR ERROR TABLE
007524  006300                                 ASL     R0
007526  006300                                 ASL     R0
007530  006300                                 ASL     R0
007532  062700  001700                 1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
```

```
2239  007536  012037  007552          MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
2240  007542  001404                  BEQ     3$              ;BRANCH IF THERE ISN'T ONE
2241  007544  104401  001167          TYPE    .$CRLF          ;"CARRIAGE RETURN - LINE FEED
2242  007550  104401                  TYPE
2243  007552  000000          2$:     .WORD   0               ;"EM" POINTER GOES HERE
2244  007554  012037  007570  3$:     MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
2245  007560  001404                  BEQ     5$              ;BRANCH IF NONE
2246  007562  104401  001167          TYPE    .$CRLF          ;CARRIAGE RETURN-LINE FEED
2247  007566  104401                  TYPE
2248  007570  000000          4$:     .WORD   0               ;"DH" POINTER GOES HERE
2249  007572  012001          5$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
2250  007574  001460                  BEQ     20$             ;BRANCH IF NONE
2251  007576  005005                  CLR     R5              ;SET INDENT SWITCH
2252  007600  012000                  MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
2253  007602  012002                  MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
2254  007604  001451                  BEQ     17$             ;BRANCH IF DH NUMBER IS 0
2255  007606  005105                  COM     R5              ;NO INDENT
2256  007610  104401  001167          TYPE    .$CRLF          ;CARRIAGE RETURN-LINE FEED
2257  007614  112003          10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
2258  007616  112004                  MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
2259  007620  006004          11$:    ROR     R4              ;OCTAL OR DECIMAL?
2260  007622  103403                  BCS     12$             ;DECIMAL--BRANCH
2261  007624  013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
2262  007626  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2263  007630  000402                  BR      13$
2264  007632          12$:
2265  007632  013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
2266  007634  104405                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2267  007636  005303          13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
2268  007640  001403                  BEQ     14$             ;NO--BRANCH
2269  007642  104401  021337          TYPE    ,_INSP          ;YES--TYPE SEPERATORS
2270  007646  000764                  BR      11$             ;LOOP
2271  007650  005302          14$:    DEC     R2              ;MORE DH'S?
2272  007652  003431                  BLE     20$             ;NO--BRANCH
2273  007654  104401  001167          TYPE    ,$CRLF          ;YES--START A NEW LINE
2274  007660  005760  000002          TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
2275  007664  001404                  BEQ     15$             ;BR IF YES - BYPASS THE INDENT
2276  007666  005105                  COM     R5              ;INDENT?
2277  007670  001002                  BNE     15$             ;NO--BRANCH
2278  007672  104401  021337          TYPE    ,LINSP          ;YES--TYPE SPACES
2279  007676  012037  007704  15$:    MOV     (R0)+,16$       ;GET NEXT DH
2280  007702  104401                  TYPE                    ;AND TYPE IT
2281  007704  000000          16$:    .WORD   0               ;DH POINTER GOES HERE
2282  007706  005710                  TST     (R0)            ;TYPE A 'DT' ?
2283  007710  001003                  BNE     21$             ;BR IF A 'DT'
2284  007712  062700  000004          ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
2285  007716  000754                  BR      14$             ;SEE IF END OF 'DF' BLOCK
2286  007720  104401  001167  21$:    TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
2287  007724  005705                  TST     R5              ;INDENT?
2288  007726  001332                  BNE     10$             ;NO--BRANCH
2289  007730  104401  021337  17$:    TYPE    ,LINSP          ;YES--TYPE SPACES
2290  007734  000727                  BR      10$             ;LOOP
2291  007736  104413          20$:    RESREG                  ;RESTORE R0-R5
2292  007740  000207                  RTS     PC              ;RETURN
2293
2294                          .SBTTL  TYPE ROUTINE
```

```
                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*
007742  105737  001157  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
007746  100002                  BPL     1$              ;;BR IF YES
007750  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
007752  000407                  BR      3$              ;;LEAVE
007754  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
007756  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
007762  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
007764  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
007766  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
007770  012600          60$:    MOV     (SP)+,R0        ;;RESTORE R0
007772  062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
007776  000002                  RTI                     ;;RETURN
010000  122716  000011  4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
010004  001430                  BEQ     8$
010006  122715  000200          CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
010012  001006                  BNE     5$
010014  005726                  TST     (SP)+           ;;POP  <CR><LF> EQUIV
010016  104401                  TYPE                    ;;TYPE A CR AND LF
010020  001167                  $CRLF
010022  105037  010156          CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
010026  000755                  BR      2$              ;;GET NEXT CHARACTER
010030  004737  010112  5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
010034  123726  001156  6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
010040  001350                  BNE     2$              ;;IF NO GO GET NEXT CHAR.
010042  013746  001154          MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                        ;;AND THE NULL CHAR.
010046  105366  000001  7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
010052  002770                  BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
010054  004737  010112          JSR     PC,$TYPEC       ;;GO TYPE A NULL
010060  105337  010156          DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
010064  000770                  BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

010066  112716  000040  8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
010072  004737  010112  9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
010076  132737  000007  010156  BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
010104  001372                  BNE     9$              ;;TAB STOP
010106  005726                  TST     (SP)+           ;;POP SPACE OFF STACK
010110  000724                  BR      2$              ;;GET NEXT CHARACTER
```

```
2351  010112  105777  171032          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2352  010116  100375                          BPL     $TYPEC
2353  010120  116677  000002  171024          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2354  010126  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2355  010134  001003                          BNE     1$              ;;BRANCH IF NO
2356  010136  105037  010156                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2357  010142  000406                          BR      $TYPEX          ;;EXIT
2358  010144  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2359  010152  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2360  010154  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2361  010156  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2362  010160  000207                  $TYPEX: RTS     PC
2363
2364
2365                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2366
2367                                   ;;**************************************************************
2368                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2369                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2370                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2371                                   ;*CALL:
2372                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2373                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
2374                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2375                                   ;*      .BYTE   M               ;;M=1 OR 0
2376                                   ;*                                      ;;1=TYPE LEADING ZEROS
2377                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
2378                                   ;*
2379                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2380                                   ;*$TYPOS OR $TYPOC
2381                                   ;*CALL:
2382                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2383                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
2384                                   ;*
2385                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2386                                   ;*CALL:
2387                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2388                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
2389
2390  010162  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
2391  010166  116637  000001  010405          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2392  010174  112637  010407                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2393  010200  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2394  010204  000406                          BR      $TYPON
2395  010206  112737  000001  010405  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2396  010214  112737  000006  010407          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2397  010222  112737  000005  010404  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2398  010230  010346                          MOV     R3,-(SP)        ;;SAVE R3
2399  010232  010446                          MOV     R4,-(SP)        ;;SAVE R4
2400  010234  010546                          MOV     R5,-(SP)        ;;SAVE R5
2401  010236  113704  010407                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2402  010242  005404                          NEG     R4
2403  010244  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2404  010250  110437  010406                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2405  010254  113704  010405                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2406  010260  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
```

```
2407  010264  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
2408  010266  006105                  1$:   RO      R5              ;;ROTATE MSB INTO "C"
2409  010270  000404                        BR      3$              ;;GO DO MSB
2410  010272  006105                  2$:   ROL     R5              ;;FORM THIS DIGIT
2411  010274  006105                        ROL     R5
2412  010276  006105                        ROL     R5
2413  010300  010503                        MOV     R5,R3
2414  010302  006103                  3$:   ROL     R3              ;;GET LSB OF THIS DIGIT
2415  010304  105337  010406                DECB    $OMODE          ;;TYPE THIS DIGIT?
2416  010310  100016                        BPL     7$              ;;BR IF NO
2417  010312  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
2418  010316  001002                        BNE     4$              ;;TEST FOR 0
2419  010320  005704                        TST     R4              ;;SUPPRESS THIS 0?
2420  010322  001403                        BEQ     5$              ;;BR IF YES
2421  010324  005204                  4$:   INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2422  010326  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2423  010332  052703  000040            5$: BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2424  010336  110337  010402                MOVB    R3,8$           ;;SAVE FOR TYPING
2425  010342  104401  010402                TYPE    ,8$             ;;GO TYPE THIS DIGIT
2426  010346  105337  010404            7$: DECB    $OCNT           ;;COUNT BY 1
2427  010352  003347                        BGT     2$              ;;BR IF MORE TO DO
2428  010354  002402                        BLT     6$              ;;BR IF DONE
2429  010356  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2430  010360  000744                        BR      2$              ;;GO DO THE LAST DIGIT
2431  010362  012605                  6$:   MOV     (SP)+,R5        ;;RESTORE R5
2432  010364  012604                        MOV     (SP)+,R4        ;;RESTORE R4
2433  010366  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2434  010370  016666  000002  000004        MOV     2(SP),4(SP      ;;SET THE STACK FOR RETURNING
2435  010376  012616                        MOV     (SP)+,(SP)
2436  010400  000002                        RTI                     ;;RETURN
2437  010402     000                  8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2438  010403     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2439  010404     000                  $OCNT:.BYTE   0               ;;OCTAL DIGIT COUNTER
2440  010405     000                  $OFILL:.BYTE  0               ;;ZERO FILL SWITCH
2441  010406  000000                  $OMODE:.WORD  0               ;;NUMBER OF DIGITS TO TYPE
2442
2443                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2444
2445                                  ;;**************************************************************
2446                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2447                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2448                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2449                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2450                                  ;*REPLACED WITH SPACES.
2451                                  ;*CALL:
2452                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
2453                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
2454
2455  010410                          $TYPDS:
2456  010410  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2457  010412  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2458  010414  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2459  010416  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2460  010420  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2461  010422  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
2462  010426  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
```

```
2463  010432  100004                          BPL     1$              ;;BR IF INPUT IS POS.
2464  010434  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
2465  010436  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
2466  010444  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
2467  010446  012703  010624                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
2468  010452  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
2469  010456  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
2470  010460  016001  010614                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
2471  010464  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
2472  010466  002402                          BLT     4$              ;;BR IF DONE
2473  010470  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2474  010472  000774                          BR      3$
2475  010474  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
2476  010476  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
2477  010500  001002                          BNE     5$              ;;FALL THROUGH IF 0
2478  010502  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
2479  010504  100407                          BMI     7$              ;;BR IF YES
2480  010506  106316                  5$:     ASLB    (SP)            ;;MSD?
2481  010510  103003                          BCC     6$              ;;BR IF NO
2482  010512  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2483  010520  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2484  010524  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2485  010530  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2486  010532  005720                          TST     (R0)+           ;;JUST INCREMENTING
2487  010534  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
2488  010540  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
2489  010542  003002                          BGT     8$              ;;GO TO EXIT
2490  010544  010502                          MOV     R5,R2           ;;GET THE LSD
2491  010546  000764                          BR      6$              ;;GO CHANGE TO ASCII
2492  010550  105726                  9$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2493  010552  100003                          BPL     9$              ;;BR IF NO
2494  010554  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2495  010562  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
2496  010564  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2497  010566  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2498  010570  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2499  010572  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2500  010574  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2501  010576  104401  010624                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2502  010602  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2503  010610  012616                          MOV     (SP)+,(SP)
2504  010612  000002                          RTI                     ;;RETURN TO USER
2505  010614  023420                  $DTBL:  10000.
2506  010616  001750                          1000.
2507  010620  000144                          100.
2508  010622  000012                          10.
2509  010624  000004                  $DBLK:  .BLKW   4
2510
2511                                  .SBTTL  TTY INPUT ROUTINE
2512
2513                                  ;;**********************************************************************
2514                                  .ENABL  LSB
2515  010634  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
2516  010636  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
2517  010640  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
2518  010642  000007                  $TKQSRT: .BLKB  7               ;;TTY KEYBOARD QUEUE
```

```
 2519            010651                  $TKQEND=.
 2520            010652                  .EVEN
 2521
 2522                                    ;*TK INITIALIZE ROUTINE
 2523                                    ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 2524                                    ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 2525                                    ;
 2526                                    ;*CALL:
 2527                                    ;*      JSR     PC,$TKINT
 2528                                    ;*      RETURN
 2529                                    ;
 2530  010652  005037  010634            $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 2531  010656  012737  010642  010636            MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 2532  010664  013737  010636  010640            MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
 2533  010672  012737  010722  000060            MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 2534  010700  012737  000200  000062            MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
 2535  010706  005777  170234                    TST     @$TKB           ;;CLEAR DONE FLAG
 2536  010712  012777  000100  170224            MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 2537  010720  000207                            RTS     PC              ;;RETURN TO CALLER
 2538
 2539                                    ;*TK SERVICE ROUTINE
 2540                                    ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 2541                                    ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 2542                                    ;*IT IN THE QUEUE.
 2543                                    ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
 2544                                    ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (@CNTLC)
 2545                                    ;
 2546  010722  117746  170220            $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 2547  010726  042716  177600                    BIC     #↑C177,(SP)     ;;STRIP THE JUNK
 2548  010732  021627  000003                    CMP     (SP),#3         ;;IS IT A CONTROL C?
 2549  010736  001007                            BNE     1$              ;;BRANCH IF NO
 2550  010740  104401  012035                    TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
 2551  010744  004737  010652                    JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 2552  010750  005726                            TST     (SP)+           ;;CLEAN UP STACK
 2553  010752  000177  170322                    JMP     @CNTLC          ;;CONTROL C RESTART
 2554  010756  021627  000007            1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 2555  010762  001004                            BNE     2$              ;;BRANCH IF NO
 2556  010764  022737  000176  001140            CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 2557  010772  001500                            BEQ     6$              ;;GO TO SWR CHANGE
 2558
 2559  010774                            2$:
 2560  010774  022737  000007  010634            CMP     #7,$TKCNT       ;;IS THE QUEUE FULL?
 2561  011002  001004                            BNE     3$              ;;BRANCH IF NO
 2562  011004  104401  001162                    TYPE    ,$BELL          ;;RING THE TTY BELL
 2563  011010  005726                            TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 2564  011012  000451                            BR      5$              ;;EXIT
 2565  011014  021627  000023            3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 2566  011020  001021                            BNE     32$             ;;BRANCH IF NO
 2567  011022  005077  170116                    CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 2568  011026  005726                            TST     (SP)+           ;;CLEAN CHAR OFF STACK
 2569  011030  105777  170110            31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 2570  011034  100375                            BPL     31$             ;;LOOP UNTIL ITS THERE
 2571  011036  117746  170104                    MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 2572  011042  042716  177600                    BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
 2573  011046  022627  000021                    CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 2574  011052  001366                            BNE     31$             ;;BRANCH IF NO
```

```
2575  011054  012777  000100  170062          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
2576  011062  000002                          RTI                     ;;RETURN
2577  011064  005237  010634          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
2578  011070  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
2579  011074  002405                          BLT     4$              ;;BRANCH IF YES
2580  011076  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
2581  011102  003002                          BGT     4$              ;;BRANCH IF YES
2582  011104  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
2583  011110  112677  177522          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
2584  011114  005237  010636                  INC     $TKQIN          ;;UPDATE THE POINTER
2585  011120  023727  010636  010651          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
2586  011126  001003                          BNE     5$              ;;BRANCH IF NO
2587  011130  012737  010642  010636          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
2588  011136  000002                  5$:     RTI                     ;;RETURN
2589
2590                                          ;;**********************************************************************
2591                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2592                                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
2593                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
2594                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
2595  011140  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
2596  011146  001124                          BNE     15$             ;;EXIT IF NOT
2597  011150  105777  167770                  TSTB    @$TKS           ;;IS A CHAR WAITING?
2598  011154  100121                          BPL     15$             ;;IF NOT, EXIT
2599  011156  117746  167764                  MOVB    @$TKB,-(SP)     ;;YES
2600  011162  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
2601  011166  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
2602  011172  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
2603                                                                  ;;AND EXIT
2604
2605                                          ;;**********************************************************************
2606                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
2607                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
2608                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
2609  011174  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
2610  011202  001674                          BEQ     2$              ;;BRANCH IF YES
2611  011204  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
2612  011206  004737  010652                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
2613  011212  005077  167726                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
2614  011216  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
2615
2616  011224  104401  012047                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
2617  011230  104401  012054          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
2618  011234  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
2619  011240  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2620  011242  104401  012065                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
2621  011246  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
2622  011250  005046                          CLR     -(SP)           ;;THE NEW SWR
2623  011252  105777  167666          7$:     TSTB    @$TKS           ;;CHAR THERE?
2624  011256  100375                          BPL     7$              ;;IF NOT TRY AGAIN
2625
2626  011260  117746  167662                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
2627  011264  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2628
2629  011270  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
2630  011274  001015                          BNE     9$              ;;BRANCH IF NOT
```

```
2631  011276  104401  012035                    TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (↑C)
2632  011302  062706  000006                    ADD     #6,SP           ;;CLEAN UP STACK
2633  011306  123727  001135  000001            CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
2634  011314  001003                            BNE     8$              ;;BRANCH IF NO
2635  011316  012777  000100  167620            MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
2636  011324  000177  167750          8$:       JMP     @CNTLC          ;;CONTROL-C RESTART
2637
2638
2639  011330  021627  000025          9$:       CMP     (SP),#25        ;;IS IT A CONTROL-U?
2640  011334  001005                            BNE     10$             ;;BRANCH IF NOT
2641  011336  104401  012042                    TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
2642  011342  062706  000006          20$:      ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
2643  011346  000737                            BR      19$             ;;LET'S TRY IT AGAIN.
2644
2645
2646  011350  021627  000015          10$:      CMP     (SP),#15        ;;IS IT A <CR>?
2647  011354  001022                            BNE     16$             ;;BRANCH IF NO
2648  011356  005766  000004                    TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2649  011362  001403                            BEQ     11$             ;;BRANCH IF YES
2650  011364  016677  000002  167546            MOV     2(SP),@SWR      ;;SAVE NEW SWR
2651  011372  062706  000006          11$:      ADD     #6,SP           ;;CLEAR UP STACK
2652  011376  104401  001167          14$:      TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
2653  011402  123727  001135  000001            CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2654  011410  001003                            BNE     15$             ;;BRANCH IF NOT
2655  011412  012777  000100  167524            MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2656  011420  000002          15$:      RTI                             ;;RETURN
2657  011422  004737  010112          16$:      JSR     PC,$TYPEC       ;;ECHO CHAR
2658  011426  021627  000060                    CMP     (SP),#60        ;;CHAR < 0?
2659  011432  002420                            BLT     18$             ;;BRANCH IF YES
2660  011434  021627  000067                    CMP     (SP),#67        ;;CHAR > 7?
2661  011440  003015                            BGT     18$             ;;BRANCH IF YES
2662  011442  042726  000060                    BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2663  011446  005766  000002                    TST     2(SP)           ;;IS THIS THE FIRST CHAR
2664  011452  001403                            BEQ     17$             ;;BRANCH IF YES
2665  011454  006316                            ASL     (SP)            ;;NO, SHIFT PRESENT
2666  011456  006316                            ASL     (SP)            ;;   CHAR OVER TO MAKE
2667  011460  006316                            ASL     (SP)            ;;   ROOM FOR NEW ONE.
2668  011462  005266  000002          17$:      INC     2(SP)           ;;KEEP COUNT OF CHAR
2669  011466  056616  177776                    BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2670  011472  000667                            BR      7$              ;;GET THE NEXT ONE
2671  011474  104401  001166          18$:      TYPE    ,$QUES          ;;TYPE ?<CR><LF>
2672  011500  000720                            BR      20$             ;;SIMULATE CONTROL-U
2673                                .DSABL  LSB
2674
2675
2676                                ;;*****************************************************************
2677                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2678                                ;*CALL:
2679                                ;*      RDCHR                           ;;GET A CHARACTER FROM THE QUEUE
2680                                ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
2681                                ;*                                      ;;WITH PARITY BIT STRIPPED OFF
2682                                ;
2683
2684  011502  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
2685  011504  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE PS
2686  011512  005066  000004                CLR     4(SP)           ;;GET READY FOR A CHARACTER
```

```
 2687  011516  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
 2688  011520  012746  011526                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 2689  011524  000002                          RTI                     ;;POP NEW PC AND PS
 2690  011526                          64$:
 2691  011526  005737  010634          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
 2692  011532  001775                          BEQ     1$
 2693  011534  005337  010634                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
 2694  011540  117766  177074  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
 2695  011546  005237  010640                  INC     $TKQOUT         ;;UPDATE THE POINTER
 2696  011552  023727  010640  010651          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
 2697  011560  001003                          BNE     2$              ;;BRANCH IF NO
 2698  011562  012737  010642  010640          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
 2699  011570  000002                  2$:     RTI                     ;;RETURN
 2700                                  ;;**************************************************************
 2701                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 2702                                  ;*CALL:
 2703                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 2704                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 2705                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 2706
 2707  011572  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 2708  011574  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
 2709  011576  012703  012026          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 2710  011602  022703  012035          2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
 2711  011606  101456                          BLOS    4$              ;;BR IF YES
 2712  011610  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 2713  011612  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
 2714  011614  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 2715  011620  001022                          BNE     5$              ;;BR IF NO
 2716  011622  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
 2717  011624  001007                          BNE     6$              ;;BR IF NO
 2718  011626  112737  000134  012024          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 2719  011634  104401  012024                  TYPE    ,9$
 2720  011640  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
 2721  011644  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
 2722  011646  020327  012026                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
 2723  011652  103434                          BLO     4$              ;;BR IF YES
 2724  011654  111337  012024                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
 2725  011660  104401  012024                  TYPE    ,9$             ;;GO TYPE
 2726  011664  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
 2727  011666  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
 2728  011670  001406                          BEQ     7$              ;;BR IF NO
 2729  011672  112737  000134  012024          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 2730  011700  104401  012024                  TYPE    ,9$
 2731  011704  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
 2732  011706  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
 2733  011712  001003                          BNE     8$              ;;BR IF NO
 2734  011714  104401  012042                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
 2735  011720  000726                          BR      1$              ;;GO START OVER
 2736  011722  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
 2737  011726  001011                          BNE     3$              ;;BRANCH IF NO
 2738  011730  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
 2739  011732  104401  001167                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
 2740  011736  104401  012026                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
 2741  011742  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
 2742  011744  104401  001166          4$:     TYPE    ,$QUES          ;;TYPE A '?'
```

```
2743  011750  000712                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
2744  011752  111337  012024        3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2745  011756  104401  012024                TYPE    ,9$
2746  011762  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2747  011766  001305                        BNE     2$              ;;LOOP IF NOT RETURN
2748  011770  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2749  011774  104401  001170                TYPE    ,$LF            ;;TYPE A LINE FEED
2750  012000  005726                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2751  012002  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2752  012004  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2753  012006  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2754  012014  012766  012026  000004        MOV     #$TTYIN,4(SP)
2755  012022  000002                        RTI                     ;;RETURN
2756  012024     000                 9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2757  012025     000                         .BYTE   0               ;;TERMINATOR
2758  012026  000007                $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
2759  012035     136  006503  000012 $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
2760  012042  052536  005015     000 $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
2761  012047     136  006507  000012 $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
2762  012054  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
2763  012062  020075     000
2764  012065     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
2765  012072  036440  000040
2766
2767                                        .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
2768
2769                                        ;;**************************************************************
2770                                        ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
2771                                        ;*LEADING NUMBERS.
2772                                        ;*CALL
2773                                        ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
2774                                        ;*      JSR     PC,@#$SUPRS
2775
2776
2777  012076  010046                $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
2778  012100  016600  000004                MOV     4(SP),R0        ;;PICKUP THE POINTER
2779  012104  105710                1$:     TSTB    (R0)            ;;TERMINATEOR?
2780  012106  001403                        BEQ     2$              ;;BR IF YES
2781  012110  122720  000060                CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
2782  012114  001773                        BEQ     1$              ;;BR IF YES
2783  012116  005300                2$:     DEC     R0              ;;BACKUP BY "1"
2784  012120  010037  012126                MOV     R0,3$           ;;SAVE FOR TYPING
2785  012124  104401                        TYPE                    ;;GO TYPE
2786  012126  000000                3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
2787  012130  012600                        MOV     (SP)+,R0        ;;RESTORE R0
2788  012132  012616                        MOV     (SP)+,(SP)      ;;RESTORE THE STACK
2789  012134  000207                        RTS     PC              ;;RETURN
2790
2791                                        .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2792
2793                                        ;;**************************************************************
2794                                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
2795                                        ;*UNSIGNED DECIMAL ASCIZ NUMBER.
2796                                        ;*CALL
2797                                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
2798                                        ;*      JSR     PC,@#$SB2D      ;;CALL
```

```
2799                                    ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
2800
2801
2802  012136  016637  000002  012166  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
2803  012144  012746  012166                  MOV     #1$,-(SP)       ;;SET POINTER
2804  012150  004737  012172                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
2805  012154  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
2806  012160  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
2807  012164  000207                          RTS     PC              ;;RETURN
2808  012166  000000  000000          1$:     .WORD   0,0
2809
2810                                    .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2811
2812                                    ;;**********************************************************************
2813                                    ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
2814                                    ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
2815                                    ;*POSITIVE.
2816                                    ;*CALL
2817                                    ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
2818                                    ;*      JSR     PC,@#$DB2D
2819                                    ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
2820                                                                    ;;IS ON THE STACK
2821
2822
2823  012172  104412                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
2824  012174  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
2825  012200  012700  012352                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
2826  012204  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
2827  012210  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
2828  012212  012202                          MOV     (R2)+,R2
2829  012214  012737  000012  012270          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
2830  012222  012704  012302                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
2831  012226  012705  012304                  MOV     #$TNPWR+2,R5
2832  012232  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
2833  012234  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
2834  012236  005602                          SBC     R2
2835  012240  161502                          SUB     (R5),R2
2836  012242  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
2837  012244  005203                          INC     R3              ;;ADD 1 TO PARTIAL
2838  012246  000772                          BR      2$              ;;LOOP
2839  012250  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
2840  012252  005502                          ADC     R2
2841  012254  062402                          ADD     (R4)+,R2
2842  012256  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
2843  012260  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
2844  012264  110320                          MOVB    R3,(R0)+        ;;SAVE IT
2845  012266  005327                          DEC     (PC)+           ;;DONE?
2846  012270  000000                  4$:     .WORD   0
2847  012272  001357                          BNE     1$              ;;BR IF NO
2848  012274  105020                          CLRB    (R0)+           ;;TERMINATOR
2849  012276  104413                          RESREG                  ;;RESTORE REGISTERS
2850  012300  000207                          RTS     PC              ;;RETURN
2851  012302  145000                  $TNPWR: 145000                  ;;1.0E09
2852  012304  035632                          35632
2853  012306  160400                          160400                  ;;1.0E08
2854  012310  002765                          2765
```

```
2855  012312  113200                          113200                  ;;1.0E07
2856  012314  000230                          230
2857  012316  041100                          041100                  ;;1.0E06
2858  012320  000017                          17
2859  012322  103240                          103240                  ;;1.0E05
2860  012324  000001                          1
2861  012326  023420                          23420                   ;;1.0E04
2862  012330  000000                          0
2863  012332  001750                          1750                    ;;1.0E03
2864  012334  000000                          0
2865  012336  000144                          144                     ;;1.0E02
2866  012340  000000                          0
2867  012342  000012                          12                      ;;1.0E01
2868  012344  000000                          0
2869  012346  000001                          1                       ;;1.0E00
2870  012350  000000                          0
2871  012352  000014                  $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
2872
2873                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
2874
2875                                  ;;*****************************************************************
2876                                  ;*SAVE R0-R5
2877                                  ;*CALL:
2878                                  ;*      SAVREG
2879                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
2880                                  ;*
2881                                  ;*TOP---(+16)
2882                                  ;* +2---(+18)
2883                                  ;* +4---R5
2884                                  ;* +6---R4
2885                                  ;* +8---R3
2886                                  ;*+10---R2
2887                                  ;*+12---R1
2888                                  ;*+14---R0
2889
2890  012366                          $SAVREG:
2891  012366  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2892  012370  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2893  012372  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2894  012374  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2895  012376  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2896  012400  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2897  012402  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
2898  012406  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
2899  012412  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
2900  012416  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
2901  012422  000002                          RTI
2902
2903                                  ;*RESTORE R0-R5
2904                                  ;*CALL:
2905                                  ;*      RESREG
2906  012424                          $RESREG:
2907  012424  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
2908  012430  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
2909  012434  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
2910  012440  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
```

```
2911  012444  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
2912  012446  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
2913  012450  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2914  012452  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2915  012454  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2916  012456  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2917  012460  000002                        RTI
2918
2919                            .SBTTL  TRAP DECODER
2920
2921                            ;;**************************************************************
2922                            ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2923                            ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2924                            ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2925                            ;*GO TO THAT ROUTINE.
2926
2927  012462  010046            $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2928  012464  016600  000002           MOV     2(SP),R0        ;;GET TRAP ADDRESS
2929  012470  005740                   TST     -(R0)           ;;BACKUP BY 2
2930  012472  111000                   MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2931  012474  006300                   ASL     R0              ;;POSITION FOR INDEXING
2932  012476  016000  012516           MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2933  012502  000200                   RTS     R0              ;;GO TO ROUTINE
2934
2935
2936                            ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2937
2938  012504  011646            $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2939  012506  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2940  012514  000002                   RTI                     ;;RESTORE THE PSW
2941
2942                            .SBTTL  TRAP TABLE
2943
2944                            ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2945                            ;*BY THE "TRAP" INSTRUCTION.
2946
2947                            ;       ROUTINE
2948                            ;       -------
2949  012516  012504            $TRPAD: .WORD   $TRAP2
2950  012520  007742                    $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2951  012522  010206                    $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2952  012524  010162                    $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2953  012526  010222                    $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2954  012530  010410                    $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
2955
2956  012532  011230                    $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
2957
2958  012534  011140                    $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
2959  012536  011502                    $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
2960  012540  011572                    $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
2961  012542  012366                    $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
2962  012544  012424                    $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
2963
2964
2965
2966                            .SBTTL  SINGLE/DUAL PORT RH70/RM01 DRIVER (REV 1.0)
```

```
                                ;COPYRIGHT (C) 1976
                                ;DIGITAL EQUIPMENT CORP.
                                ;MAYNARD, MA 01754
                                ;AUTHOR(S): JIM LACEY/CHUCK HESS/AL BURNESS

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

                                ;STORAGE FOR RMDS, RMER1, RMER2, AND RMMR2 ON AN ERROR "2"
                                        ;RMERRS   = RMDS
                                        ;RMERRS+2 = RMER1
                                        ;RMERRS+4 = RMER2
                                        ;RMERRS+6 = RMMR2

012546  000000  000000  000000  RMERRS: .WORD   0,0,0,0
012554  000000

                                ;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)
                                        ;DRVACT=0 IF DRIVE IS IDLE
                                        ;DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
                                        ;DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION

012556  000                     DRVACT: .BYTE   0               ;DRIVE 0
012557  000                             .BYTE   0               ;DRIVE 1
012560  000                             .BYTE   0               ;DRIVE 2
012561  000                             .BYTE   0               ;DRIVE 3
012562  000                             .BYTE   0               ;DRIVE 4
012563  000                             .BYTE   0               ;DRIVE 5
012564  000                             .BYTE   0               ;DRIVE 6
012565  000                             .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
                                        ;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXISTENT
                                        ;DRVSTA>0 IF DRIVE IS ONLINE
                                        ;DRVSTA<0 IF DRIVE IS UNSAFE

012566  000                     DRVSTA: .BYTE   0               ;DRIVE 0
012567  000                             .BYTE   0               ;DRIVE 1
012570  000                             .BYTE   0               ;DRIVE 2
012571  000                             .BYTE   0               ;DRIVE 3
012572  000                             .BYTE   0               ;DRIVE 4
012573  000                             .BYTE   0               ;DRIVE 5
012574  000                             .BYTE   0               ;DRIVE 6
012575  000                             .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
                                        ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
                                        ;DRVTYP=4 IF DRIVE IS RM01
                                        ;DRVTYP=-1 IF NOT RM01

012576  000                     DRVTYP: .BYTE   0               ;DRIVE 0
012577  000                             .BYTE   0               ;DRIVE 1
012600  000                             .BYTE   0               ;DRIVE 2
012601  000                             .BYTE   0               ;DRIVE 3
012602  000                             .BYTE   0               ;DRIVE 4
012603  000                             .BYTE   0               ;DRIVE 5
```

```
3023  012604  000                         .BYTE   0               ;DRIVE 6
3024  012605  000                         .BYTE   0               ;DRIVE 7
3025
3026                                  ;TABLE OF DUAL PORT INITIALIZATION INDICATORS
3027                                          ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
3028                                          ;DPINT<0 IF INITIALIZATION IS IN PROGRESS
3029
3030  012606  000                   DPINT:  .BYTE   0               ;DRIVE 0
3031  012607  000                           .BYTE   0               ;DRIVE 1
3032  012610  000                           .BYTE   0               ;DRIVE 2
3033  012611  000                           .BYTE   0               ;DRIVE 3
3034  012612  000                           .BYTE   0               ;DRIVE 4
3035  012613  000                           .BYTE   0               ;DRIVE 5
3036  012614  000                           .BYTE   0               ;DRIVE 6
3037  012615  000                           .BYTE   0               ;DRIVE 7
3038
3039                                  ;TABLE OF PENDING DUAL PORT REQUESTS
3040                                          ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
3041                                          ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE
3042
3043  012616  000                   DPRQS:  .BYTE   0               ;DRIVE 0
3044  012617  000                           .BYTE   0               ;DRIVE 1
3045  012620  000                           .BYTE   0               ;DRIVE 2
3046  012621  000                           .BYTE   0               ;DRIVE 3
3047  012622  000                           .BYTE   0               ;DRIVE 4
3048  012623  000                           .BYTE   0               ;DRIVE 5
3049  012624  000                           .BYTE   0               ;DRIVE 6
3050  012625  000                           .BYTE   0               ;DRIVE 7
3051
3052                                  ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
3053                                          ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
3054                                          ;"DPB" OF THE I/O OPERATION.
3055
3056  012626  000000                TRNSWT: .WORD   0
3057
3058                                  ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
3059                                          ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
3060                                          ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
3061                                          ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
3062                                          ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
3063
3064  012630  000000                SRCHWT: .WORD   0
3065
3066                                  ;RM01 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
3067                                          ;ACTDRV=0 IF DRIVER IS INACTIVE
3068                                          ;ACTDRV>0 IF DRIVER IS ACTIVE
3069
3070  012632  000                   ACTDRV: .BYTE   0
3071
3072                                  ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
3073                                          ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
3074                                          ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
3075
3076  012633  000                   ACTSTR: .BYTE   0
3077
3078                                  ;UNLOAD FLAG (JLDFLG=8 BYTES)
```

```
                                        ;ULDFLG=0 IF NO UNLOAD COMMAND
                                        ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
                                        ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEJE

012634  000             ULDFLG: .BYTE   0               ;DRIVE 0
012635  000                     .BYTE   0               ;DRIVE 1
012636  000                     .BYTE   0               ;DRIVE 2
012637  000                     .BYTE   0               ;DRIVE 3
012640  000                     .BYTE   0               ;DRIVE 4
012641  000                     .BYTE   0               ;DRIVE 5
012642  000                     .BYTE   0               ;DRIVE 6
012643  000                     .BYTE   0               ;DRIVE 7

                        ;LOOK AHEAD COUNT (LACNT=8 BYTES)
                                ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED

012644  000             LACNT:  .BYTE   0               ;DRIVE 0
012645  000                     .BYTE   0               ;DRIVE 1
012646  000                     .BYTE   0               ;DRIVE 2
012647  000                     .BYTE   0               ;DRIVE 3
012650  000                     .BYTE   0               ;DRIVE 4
012651  000                     .BYTE   0               ;DRIVE 5
012652  000                     .BYTE   0               ;DRIVE 6
012653  000                     .BYTE   0               ;DRIVE 7

                        ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
                                ;SAVEFG <0 IF SAVE THE RH70/RM01 REGISTERS WHEN THE
                                ;OPERATION IS COMPLETED AS PER (DPB+14).
                                ;SAVEFG=0 IF SAVE THE RH70/RM01 REGISTERS, AS PER
                                ;(DPB+14), AFTER AN ERROR.

012654  000000          SAVEFG: .WORD   0

                        ;SEEK FLAG (SEEKFG=1 WORD)
                                ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
                                ;FOR A DATA TRANSFER START A SEARCH COMMAND
                                ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
                                ;DISREGARD THE WINDOW

012656  000000          SEEKFG: .WORD   0

                        ;TIMEOUT TABLE (TIMER=8 WORDS)
                                ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION

012660  177777          TIMER:  .WORD   -1      ;DRIVE 0
012662  177777                 .WORD   -1      ;DRIVE 1
012664  177777                 .WORD   -1      ;DRIVE 2
012666  177777                 .WORD   -1      ;DRIVE 3
012670  177777                 .WORD   -1      ;DRIVE 4
012672  177777                 .WORD   -1      ;DRIVE 5
012674  177777                 .WORD   -1      ;DRIVE 6
012676  177777                 .WORD   -1      ;DRIVE 7

                        ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
                                ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
                                ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
```

```
3135
3136  012700  177777                DTUW:   .WORD   -1
3137
3138                                  ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
3139                                          ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
3140                                          ;ATTENTION BIT
3141
3142  012702  000001                ATABIT: .BYTE   1       ;DRIVE 0
3143  012703  000002                        .BYTE   2       ;DRIVE 1
3144  012704  000004                        .BYTE   4       ;DRIVE 2
3145  012705  000010                        .BYTE   10      ;DRIVE 3
3146  012706  000020                        .BYTE   20      ;DRIVE 4
3147  012707  000040                        .BYTE   40      ;DRIVE 5
3148  012710  000100                        .BYTE   100     ;DRIVE 6
3149  012711  000200                        .BYTE   200     ;DRIVE 7
3150
3151                                  ;FSRM01 TO RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
3152                                  ;CALLING IT FATAL (MCPEMX=1 WORD)
3153
3154  012712  000000                MCPEMX: .WORD   0
3155
3156                                  ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH70/RM01),
3157                                  ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).
3158
3159  012714  176700                RMADR:  .WORD   176700
3160  012716  000254  000240        RMVEC:  .WORD   254,5*32.
3161
3162                                  ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
3163
3164  012722  000004                MXLACT: .WORD   4
3165                                  ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
3166
3167  012724  001000                MXDLTA: .WORD   8.*64.
3168                                  ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
3169
3170  012726  000200                MNDLTA: .WORD   2*64.
3171                                  ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
3172
3173  012730  000005                MXWNDW: .WORD   5
3174
3175                                  ;DEFINITIONS OF THE RH70/RM01 ADDRESS INDEXES
3176
3177          000000                RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
3178          000002                RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
3179          000004                RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
3180          000006                RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
3181          000010                RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
3182          000012                RMDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
3183          000014                RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
3184          000016                RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
3185          000020                RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
3186          000022                RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
3187          000024                RMMR1=24                ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
3188          000026                RMDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
3189          000030                RMSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
3190          000032                RMOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
```

```
                    000034                  RMDC=34                         ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
                    000036                  RMHR=36                         ;DUMMY  ADDRESS REGISTER (DRIVE REG. 13)
                    000040                  RMMR2=40                        ;MAINTENANCE REGISTER #2
                    000042                  RMER2=42                        ;ERROR REGISTER #2 (DRIVE REG. 15)
                    000044                  RMEC1=44                        ;ECC POSITION REGISTER (DRIVE REG. 16)
                    000046                  RMEC2=46                        ;ECC PATTERN REGISTER (DRIVE REG.  17)

                                            ;RH70/RM01 DRIVER INITIALIZATION CODE
                                                    ;THIS ROUTINE WILL DETERMINE WHICH RM01 DRIVES ARE
                                                    ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                                    ;TO THE PROPER STATE FOR EACH DRIVE.
                                                    ;NOTE: THIS ROUTINE CALLS DRVINT
                                            ;
                                            ;CALL
                                            ;
                                            ;       JSR     PC,RMINIT
                                            ;       RETURN
                                            ;
                                            ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                            ;
012732  104412                  RMINIT:    SAVREG                          ;SAVE R0 - R5
012734  013746  177776                      MOV     @#PS,-(SP)              ;SAVE THE PRESENT PROCESSOR STATUS
012740  012737  000240  177776              MOV     #<5*32.>,@#PS           ;CHANGE THE PRIORITY TO 5
012746  004737  020706                      JSR     PC,CLRQUE               ;CLEAR ALL REQUEST QUEUES
012752  012701  012546                      MOV     #RMERRS,R1              ;FIRST ADDRESS TO BE CLEARED
012756  012702  012656                      MOV     #SEEKFG,R2              ;LAST ADDRESS TO BE CLEARED
012762  005021                  1$:         CLR     (R1)+                   ;CLEAR
012764  020102                              CMP     R1,R2                   ;ARE WE DONE?
012766  101775                              BLOS    1$                      ;BRANCH IF NO
012770  012702  012700                      MOV     #DTUW,R2                ;LAST ADDRESS
012774  012721  177777          2$:         MOV     #-1,(R1)+               ;INITIALIZE
013000  020102                              CMP     R1,R2                   ;DONE?
013002  101774                              BLOS    2$                      ;LOOP IF NO
013004  005037  012566                      CLR     DRVSTA                  ;SET ALL DRIVES TO OFFLINE
013010  005037  012570                      CLR     DRVSTA+2
013014  005037  012572                      CLR     DRVSTA+4
013020  005037  012574                      CLR     DRVSTA+6
013024  013703  012716                      MOV     RMVEC,R3                ;SETUP THE RH70/RM01 VECTOR
013030  012723  015560                      MOV     #ISR,(R3)+
013034  013713  012720                      MOV     RMVEC+2,(R3)
013040  013704  012714                      MOV     RMADR,R4                ;FIRST ADDRESS OF RH70/RM01
013044  012764  000040  000010              MOV     #BIT05,RMCS2(R4)        ;MASSBUS INIT
013052  005001                              CLR     R1                      ;START WITH DRIVE 0
013054  004037  013144          3$:         JSR     R0,DRVINT               ;INIT THE DRIVE
013060  000401                              BR      4$                      ;'D/A' NOT SET OR PARITY ERROR
013062  000402                              BR      5$                      ;NORMAL RETURN
013064  105061  012566          4$:         CLRB    DRVSTA(R1)              ;SET DRIVE STATUS TO OFFLINE
013070  005201                  5$:         INC     R1                      ;GO TO NEXT DRIVE
013072  042701  177770                      BIC     #^C7,R1                 ;MASK OUT UNUSED BITS
013076  001366                              BNE     3$                      ;BR IF MORE DRIVES TO GO
013100  012701  000007                      MOV     #7,R1                   ;START WITH DRIVE 7
013104  005037  177776                      CLR     @#PS                    ;CLEAR THE PROCESSOR STATUS
013110  105761  012606          6$:         TSTB    DPINT(R1)               ;WAITING FOR DRIVE TO SWITCH PORTS ?
013114  001405                              BEQ     8$                      ;BR NOT WAITING
013116  004737  020342                      JSR     PC,SET.IE               ;SET INTERRUPT
013122  105761  012606          7$:         TSTB    DPINT(R1)               ;DRIVE SWITCHED PORTS ?
```

```
3247  013126  001375                        BNE     7$              ;BR IF NOT
3248  013130  005301                8$:     DEC     R1              ;GO TO THE NEXT DRIVE
3249  013132  100366                        BPL     6$              ;CHECK NEXT DRIVE
3250  013134  012637  177776                MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
3251  013140  104413                        RESREG                  ;RESTORE R0 - R5
3252  013142  000207                        RTS     PC              ;BYE-BYE
3253
3254                                ;DRIVE INITILIZATION ROUTINE
3255                                        ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
3256                                        ;AN RMO1. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
3257                                        ;IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
3258                                        ;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
3259                                        ;DRVSTA IS SET TO THE PROPER CONDITION.
3260                                ;CALL
3261                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
3262                                ;       MOV     RMADR,R4        ;UNIBUS ADDRESS OF RH70/RMO1 (RMCS1)
3263                                ;       JSR     R0,DRVINT       ;CALLED BY A JSR
3264                                ;       RETURN1                 ;ERROR OCCURRED (PARITY)
3265                                ;       RETURN2                 ;NORMAL RETURN
3266                                ;
3267
3268  013144  010546                DRVINT: MOV     R5,-(SP)        ;SAVE R5
3269  013146  105061  012566                CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
3270  013152  105061  012576                CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
3271  013156  105061  012634                CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
3272  013162  010164  000010                MOV     R1,RMCS2(R4)    ;SELECT A DRIVE
3273  013166  112764  000111  000000        MOVB    #111,RMCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
3274  013174  032764  010000  000010        BIT     #BIT12,RMCS2(R4) ;NONEXISTENT DRIVE?
3275  013202  001403                        BEQ     1$              ;NO---BRANCH
3276  013204  004737  020342                JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
3277  013210  000476                        BR      6$              ;LEAVE THIS ROUTINE
3278  013212  105061  012566        1$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
3279  013216  032764  004000  000000        BIT     #BIT11,RMCS1(R4)  ;SEE IF DRIVE AVAILABLE
3280  013224  001472                        BEQ     7$              ;BR IF DRIVE NOT AVAILABLE
3281  013226  004037  017652                JSR     R0,RD.RM        ;READ THE DRIVE TYPE REG.
3282  013232  000026                        RMDT
3283  013234  013432                        8$                      ;ERROR RETURN ADDRESS
3284  013236  012605                        MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
3285  013240  112761  000004  012576        MOVB    #4,DRVTYP(R1)   ;SET RMO1 INDICATOR
3286  013246  022705  020024                CMP     #20024,R5       ;SINGLE PORT RMO1 ?
3287  013252  001407                        BEQ     2$              ;BR IF YES
3288  013254  022705  024024                CMP     #24024,R5       ;DUAL PORT RMO1 ?
3289  013260  001404                        BEQ     2$              ;BR IF YES
3290  013262  112761  177777  012576        MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
3291  013270  000446                        BR      6$              ;EXIT
3292  013272  012746  000121        2$:     MOV     #121,-(SP)      ;DO A "READ-IN PRESET"
3293  013276  004037  020032                JSR     R0,WRT.RM
3294  013302  000000                        RMCS1
3295  013304  013432                        8$
3296  013306  012746  010000                MOV     #BIT12,-(SP)    ;SET FMT22=1
3297  013312  004037  020032                JSR     R0,WRT.RM
3298  013316  000032                        RMOF
3299  013320  013432                        8$
3300  013322  004037  017652                JSR     R0,RD.RM        ;READ RMDS
3301  013326  000012                        RMDS
3302  013330  013432                        8$
```

```
3303  013332  012605                        MOV     (SP)+,R5;       ;AND SAVE IT IN R5
3304  013334  100015                        BPL     4$              ;BRANCH IF ATA=0
3305  013336  116164  012702  000016        MOVB    ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
3306  013344  004037  017652                JSR     R0,RD.RM        ;FIND OUT WHY ATA=1
3307  013350  000014                        RMER1
3308  013352  013432                        8$
3309  013354  006126                        ROL     (SP)+           ;IS IT UNSAFE?
3310  013356  100004                        BPL     4$              ;BR IF NOT
3311  013360  112761  177777  012566        MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
3312  013366  000407                        BR      6$              ;EXIT
3313  013370  005105                4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
3314  013372  042705  167077                BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
3315  013376  001003                        BNE     6$              ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
3316  013400  112761  000001  012566        MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
3317  013406  005720                6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
3318  013410  000410                        BR      8$              ;EXIT
3319  013412  006301                7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
3320  013414  012761  003720  012660        MOV     #2000.,TIMER(R1)  ;START 2 SEC TIMER
3321  013422  006201                        ASR     R1              ;RESTORE R1
3322  013424  112761  177777  012606        MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INIDICATOR
3323  013432  012605                8$:     MOV     (SP)+,R5        ;RESTORE R5
3324  013434  000200                        RTS     R0              ;EXIT
3325
3326                                  ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
3327                                  ;
3328                                  ;CALL
3329                                  ;
3330                                  ;       JSR     R0,@#RM01       ;CALL THE RM01 DRIVER
3331                                  ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
3332                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
3333                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
3334                                                                  ;IS AN ERROR CONDITION
3335
3336  013436  013746  177776        RM01:   MOV     @#PS,-(SP)      ;SAVE THE CALLING STATUS
3337  013442  013737  012720  177776        MOV     RMVEC+2,@#PS    ;DON'T ALLOW ANY RM01 INTERRUPTS
3338  013450  112737  000001  012632        MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
3339  013456  104412                        SAVREG                  ;SAVE R0 - R5
3340  013460  011002                        MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
3341  013462  005062  000016                CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
3342  013466  111201                        MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
3343  013470  013704  012714                MOV     RMADR,R4        ;UNIBUS ADDRESS OF RMCS1
3344  013474  105761  012566                TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
3345  013500  003014                        BGT     1$              ;BRANCH IF ONLINE
3346  013502  105761  012634                TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
3347  013506  001036                        BNE     3$              ;BRANCH IF YES
3348  013510  105761  012606                TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
3349  013514  001042                        BNE     5$              ;BR IF YES
3350  013516  004037  013144                JSR     R0,DRVINT       ;GO INIT. THE DRIVE
3351  013522  000434                        BR      4$              ;ERROR RETURN
3352  013524  105761  012566                TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
3353  013530  003445                        BLE     6$              ;BR IF NOT
3354  013532  105761  012616        1$:     TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
3355  013536  001031                        BNE     5$              ;BR IF YES
3356  013540  010164  000010                MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
3357  013544  004037  021004                JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
3358  013550  000460                        BR      9$              ;QUEUE IS FULL
```

```
3359  013552  122762  000103  000002  .       CMPB    #103,2(R2)          ;IS THIS REQ. FOR AN UNLOAD?
3360  013560  001003                          BNE     2$                  ;BR IF NO
3361  013562  112761  177777  012634          MOVB    #-1,ULDFLG(R1)      ;SET THE "UNLOAD IN QUEUE" FLAG
3362  013570  105761  012556          2$:     TSTB    DRVACT(R1)          ;IS THIS DRIVE ACTIVE?
3363  013574  001043                          BNE     8$                  ;BR IF YES
3364  013576  004737  013730                  JSR     PC,OPT              ;CALL THE OPTIMIZER
3365  013602  000440                          BR      8$
3366  013604  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
3367  013612  000434                          BR      8$                  ;EXIT
3368  013614  004737  015050          4$:     JSR     PC,CI7              ;GO HANDLE THE PARITY ERROR
3369  013620  000431                          BR      8$
3370  013622  004037  021004          5$:     JSR     R0,DRVQUE           ;PUT REQUEST IN QUEUE
3371  013626  000431                          BR      9$                  ;QUEUE IS FULL
3372  013630  032714  000100                  BIT     #BIT06,(R4)         ;IS 'IE' SET ALREADY ?
3373  013634  001023                          BNE     8$                  ;BR IF IT IS
3374  013636  004737  020342                  JSR     PC,SET.IE           ;SET INTERRUPT
3375  013642  000420                          BR      8$                  ;RETURN, REQUEST IN QUEUE
3376  013644  105761  012566          6$:     TSTB    DRVSTA(R1)          ;SEE IF DRIVE OFFLINE OR UNSAFE
3377  013650  002412                          BLT     7$                  ;BR IF UNSAFE
3378  013652  012762  140000  000016          MOV     #BIT15!BIT14,16(R2) ;SET OFFLINE ERROR INDICATOR
3379  013660  105761  012576                  TSTB    DRVTYP(R1)          ;SEE IF OFFLINE OR NONEXISTENT
3380  013664  001007                          BNE     8$                  ;BR IF OFFLINE
3381  013666  012762  100002  000016          MOV     #BIT15!BIT01,16(R2) ;REPORT DRIVE NONEXISTENT
3382  013674  000403                          BR      8$                  ;GO TO EXIT
3383  013676  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2) ;DRIVE IS UNSAFE
3384  013704  104413                  8$:     RESREG                      ;RESTORE R0 - R5
3385  013706  005720                          TST     (R0)+               ;SETUP FOR NORMAL RETURN
3386  013710  000401                          BR      10$                 ;FINISH UP, THEN EXIT
3387  013712  104413                  9$:     RESREG                      ;RESTORE R0 - R5
3388  013714  005720                  10$:    TST     (R0)+               ;CORRECT THE RETURN ADDRESS
3389  013716  105037  012632                  CLRB    ACTDRV              ;CLEAR "ACTIVE DRIVER" FLAG
3390  013722  012637  177776                  MOV     (SP)+,@#PS          ;RETURN "PS" TO USER LEVEL
3391  013726  000200                          RTS     R0                  ;RETURN TO CALLER
3392
3393                                  ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
3394                                  ;
3395                                  ;CALL
3396                                  ;       MOV     #DRVNUM,R1          ;DRIVE NUMBER TO R1
3397                                  ;       JSR     PC,OPT              ;SETUP A COMMAND
3398                                  ;
3399  013730  104412                  OPT:    SAVREG                      ;SAVE R0 - R5
3400  013732  013746  177776                  MOV     @#PS,-(SP)          ;SAVE PROC. STATUS
3401  013736  004737  021060                  JSR     PC,GETREQ           ;GET "DPB" POINTER OF REQUEST
3402  013742  005702                          TST     R2                  ;IS THERE A REQUEST IN QUEUE?
3403  013744  001511                          BEQ     7$                  ;NO--BRANCH TO EXIT
3404  013746  032764  010000  000012          BIT     #BIT12,RMDS(R4)     ;IS MOL STILL SET ?
3405  013754  001404                          BEQ     9$                  ;BR IF NOT
3406  013756  032764  000100  000012          BIT     #BIT6,RMDS(R4)      ;IS VV SET ?
3407  013764  001003                          BNE     10$                 ;BR IF IT IS
3408  013766  004037  013144          9$:     JSR     R0,DRVINT           ;SEE IF DRIVE STILL ONLINE ?
3409  013772  000474                          BR      6$                  ;PARITY OR 'DVA' NOT SET
3410  013774  105761  012566          10$:    TSTB    DRVSTA(R1)          ;IS DRIVE ONLINE?
3411  014000  003014                          BGT     1$                  ;YES--BRANCH
3412  014002  004737  021102                  JSR     PC,POPQUE           ;NO--REMOVE REQUEST FROM QUEUE
3413  014006  012762  140000  000016          MOV     #BIT15!BIT14,16(R2) ;SET OFFLINE STATUS ERROR INDICATOR
3414  014014  105761  012566                  TSTB    DRVSTA(R1)          ;IS DRIVE UNSAFE ?
```

```
3415  014020  100070                        BPL     8$                ;BR TO EXIT IF NOT
3416  014022  012762  110000  000016        MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
3417  014030  000464                        BR      8$                ;BRANCH TO EXIT
3418  014032  012746  000111          1$:   MOV     #111,-(SP)        ;LOAD COMMAND ONTO THE STACK
3419  014036  004037  020032                JSR     R0,WRT.RM         ;LOAD THE REGISTER
3420  014042  000000                        RMCS1                     ;REGISTER INCREMENT
3421  014044  014164                        6$                        ;ERROR RETURN ADDRESS
3422  014046  032714  004000                BIT     #BIT11,(R4)       ;DRIVE AVAILABLE ?
3423  014052  001433                        BEQ     5$                ;BR IF NOT
3424  014054  122762  000150  000002        CMPB    #150,2(R2)        ;IS THE REQUEST FOR I/O?
3425  014062  002403                        BLT     2$                ;YES--BRANCH
3426  014064  004737  014434                JSR     PC,CI4            ;CALL THE COMMAND INITIATOR
3427  014070  000444                        BR      8$                ;BRANCH TO EXIT
3428  014072  005737  012700          2$:   TST     DTUW              ;DATA TRANSFER UNDERWAY?
3429  014076  002016                        BGE     4$                ;YES--GO START A SEARCH
3430  014100  136137  012702  012630        BITB    ATABIT(R1),SRCHWT ;IMPLIED SEEK SET ?
3431  014106  001407                        BEQ     3$                ;NO
3432  014110  005737  012656                TST     SEEKFG            ;DO IMPLIED SEEKS?
3433  014114  100404                        BMI     3$                ;YES---BRANCH
3434  014116  004037  015404                JSR     R0,LA             ;NO--DO LOOK AHEAD
3435  014122  000427                        BR      8$                ;RETURN HERE ON A PARITY ERROR
3436  014124  000403                        BR      4$                ;GO START A SEARCH
3437  014126  004737  014212          3$:   JSR     PC,CI1            ;START A DATA TRANSFER
3438  014132  000423                        BR      8$
3439  014134  004737  014326          4$:   JSR     PC,CI3            ;START A SEARCH
3440  014140  000420                        BR      8$                ;GO TO THE EXIT
3441  014142  112761  177777  012616  5$:   MOVB    #-1,DPRQS(R1)     ;SET PORT REQUEST INDICATOR
3442  014150  010103                        MOV     R1,R3             ;SET UP TO ADDRESS WORDS
3443  014152  006303                        ASL     R3                ;CONVERT TO WORD INDEX
3444  014154  012763  023420  012660        MOV     #10000.,TIMER(R3)   ;START 10 SEC TIMER
3445  014162  000402                        BR      7$                ;EXIT
3446  014164  004737  015050          6$:   JSR     PC,CI7            ;PROCESS THE PARITY ERROR
3447  014170  032714  000100          7$:   BIT     #BIT06,(R4)       ;SEE IF 'IE' ALREADY SET
3448  014174  001002                        BNE     8$                ;BR IF SET
3449  014176  004737  020342                JSR     PC,SET.IE         ;SET "IE" WITHOUT A "TRE"
3450  014202  012637  177776          8$:   MOV     (SP)+,@#PS        ;RESTORE PROC. STATUS
3451  014206  104413                        RESREG                    ;RESTORE R0 - R5
3452  014210  000207                        RTS     PC
3453
3454                                  ;COMMAND INITIATOR
3455                                  ;
3456                                  ;CALL
3457                                  ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER
3458                                  ;       MOV     #DPB,R2           ;ADDRESS OF DPB
3459                                  ;       JSR     PC,CI?            ;CI?= CI1,CI3, OR CI4
3460                                  ;                                 ;WHERE:
3461                                  ;                                 ;CI1=DATA TRANSFER
3462                                  ;                                 ;CI2=SEARCH REQUESTED BY DATA XFER
3463                                  ;                                 ;CI4=NOT DATA TRANSFER
3464
3465  014212  004737  021102          CI1:   JSR     PC,POPQUE         ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
3466  014216  146137  012702  012630        BICB    ATABIT(R1),SRCHWT ;CLEAR THE IMPLIED SEEK SET
3467  014224  010237  012626                MOV     R2,TRNSWT         ;PUT REQ. IN TRANSFER WAIT QUEUE
3468  014230  010203                        MOV     R2,R3             ;DPB ADDRESS TO R3
3469  014232  013704  012714                MOV     RMADR,R4          ;RMCS1 ADDRESS
3470  014236  010164  000010                MOV     R1,RMCS2(R4)      ;SELECT DRIVE
```

```
3471  014242  062703  000004                ADD     #4,R3           ;DESIRED WORD COUNT
3472  014246  062704  000002                ADD     #2,R4           ;RMWC ADDRESS
3473  014252  012324                       MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
3474  014254  012324                       MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
3475  014256  012346                       MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
3476  014260  004037  020032               JSR     R0,WRT.RM       ;CALL THE LOAD(WRITE) ROUTINE
3477  014264  000006                       RMDA                    ;INDEX OF REGISTER TO LOAD
3478  014266  015050                       CI7                     ;ERROR RETURN ADDRESS
3479  014270  012346                       MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
3480  014272  004037  020032               JSR     R0,WRT.RM
3481  014276  000034                       RMDC
3482  014300  015050                       CI7
3483  014302  016246  000002               MOV     2(R2),-(SP)     ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
3484  014306  004037  020032               JSR     R0,WRT.RM
3485  014312  000000                       RMCS1
3486  014314  015050                       CI7
3487  014316  010137  012700               MOV     R1,DTUW         ;SET "DATA TRANSFER UNDERWAY"
3488  014322  000137  015012               JMP     CI5
3489  014326  013704  012714       CI3:    MOV     RMADR,R4        ;RMCS1 ADDRESS
3490  014332  010164  000010               MOV     R1,RMCS2(R4)    ;SELECT DRIVE
3491  014336  016246  000012               MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
3492  014342  004037  020032               JSR     R0,WRT.RM
3493  014346  000034                       RMDC
3494  014350  015050                       CI7
3495  014352  116203  000010               MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
3496  014356  163703  012730               SUB     MXWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
3497  014362  002002                       BGE     1$
3498  014364  062703  000040               ADD     #32.,R3
3499  014370  010346               1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
3500  014372  116266  000011  000001       MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
3501  014400  004037  020032               JSR     R0,WRT.RM       ;LOAD DESIRED TRACK & SECTOR
3502  014404  000006                       RMDA
3503  014406  015050                       CI7
3504  014410  012746  000131               MOV     #131,-(SP)      ;START A SEARCH
3505  014414  004037  020032               JSR     R0,WRT.RM
3506  014420  000000                       RMCS1
3507  014422  015050                       CI7
3508  014424  156137  012702  012630       BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
3509  014432  000567                       BR      CI5
3510  014434  013704  012714       CI4:    MOV     RMADR,R4        ;RMCS1 ADDRESS
3511  014440  010164  000010               MOV     R1,RMCS2(R4)    ;SELECT DRIVE
3512  014444  116203  000002               MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
3513  014450  122703  000131               CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
3514  014454  001007                       BNE     1$              ;BRANCH IF NO
3515  014456  016246  000010               MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
3516  014462  004037  020032               JSR     R0,WRT.RM
3517  014466  000006                       RMDA
3518  014470  015050                       CI7
3519  014472  000403                       BR      2$              ;GO LOAD CYLINDER
3520  014474  122703  000105       1$:     CMPB    #105,R3         ;IS IT A SEEK COMMAND
3521  014500  001007                       BNE     3$              ;BRANCH IF NO
3522  014502  016246  000012       2$:     MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
3523  014506  004037  020032               JSR     R0,WRT.RM
3524  014512  000034                       RMDC
3525  014514  015050                       CI7
3526  014516  000546                       BR      CI6
```

```
3527  014520  122703  000115          3$:   CMPB   #115,R3          ;IS IT AN "OFFSET" COMMAND?
3528  014524  001013                        BNE    4$               ;BR IF NO
3529  014526  004037  017652                JSR    R0,RD.RM         ;MERGE THE OFFSET VALUE INTC RMOF
3530  014532  000032                        RMOF                    ;BUT DON'T CHANGE THE UPPER
3531  014534  015050                        CI7
3532  014536  116216  000001                MOVB   1(R2),(SP)       ;BYTE WHEN LOADING THE
3533  014542  004037  020032                JSR    R0,WRT.RM        ;REGISTER (RMOF)
3534  014546  000032                        RMOF
3535  014550  015050                        CI7
3536  014552  000530                        BR     CI6              ;GO START THE COMMAND
3537  014554  122703  000107          4$:   CMPB   #107,R3          ;IS IT A "RECALIBRATE" COMMAND?
3538  014560  001525                        BEQ    CI6              ;BRANCH IF YES
3539  014562  122703  000117                CMPB   #117,R3          ;IS IT A RETURN TO CENTER?
3540  014566  001522                        BEQ    CI6              ;BRANCH IF YES
3541  014570  122703  000103                CMPB   #103,R3          ;IS IT AN "UNLOAD" COMMAND?
3542  014574  001016                        BNE    5$               ;BRANCH IF NO
3543  014576  112761  000001  012556        MOVB   #1,DRVACT(R1)    ;SET THE DRIVE ACTIVE INDICATOR
3544  014604  105061  012566                CLRB   DRVSTA(R1)       ;PUT DRIVE STATUS TO OFFLINE
3545  014610  112761  000001  012634        MOVB   #1,ULDFLG(R1)    ;SET "UNLOAD IN PROGRESS" FLAG
3546  014616  010346                        MOV    R3,-(SP)         ;START THE "UNLOAD" COMMAND
3547  014620  004037  020032                JSR    R0,WRT.RM
3548  014624  000000                        RMCS1
3549  014626  015050                        CI7
3550  014630  000207                        RTS    PC               ;RETURN TO USER
3551  014632  122703  000143          5$:   CMPB   #143,R3          ;IS IT A "SET FORMAT" COMMAND?
3552  014636  001014                        BNE    6$               ;BRANCH IF NO
3553  014640  004037  017652                JSR    R0,RD.RM         ;READ THE OFFSET REGISTER
3554  014644  000032                        RMOF
3555  014646  015050                        CI7
3556  014650  116266  000001  000001        MOVB   1(R2),1(SP)      ;COMBINE "FMT22","ECI", AND "HCI"
3557  014656  004037  020032                JSR    R0,WRT.RM        ;LOAD "FMT22", "ECI", AND/OR "HCI".
3558  014662  000032                        RMOF
3559  014664  015050                        CI7
3560  014666  000436                        BR     12$
3561  014670  122703  000141          6$:   CMPB   #141,R3          ;IS IT A "GET REGISTER" COMMAND?
3562  014674  001023                        BNE    10$              ;BRANCH IF NO
3563  014676  016203  000006          7$:   MOV    6(R2),R3         ;POINTS TO 1ST ADDRESS OF WHERE
3564                                                                ;TO PUT THE REGISTER(S)
3565  014702  116237  000010  014720        MOVB   10(R2),9$        ;INIT. THE INDEX FOR THE FIRST REG.
3566  014710  116205  000011                MOVB   11(R2),R5        ;INDEX OF LAST REG. TO MOVE
3567  014714  004037  017652          8$:   JSR    R0,RD.RM         ;READ RH70/RM01 REGISTER
3568  014720  000000          9$:   RMCS1                           ;INDEX OF REG. TO READ
3569  014722  015050                        CI7
3570  014724  012623                        MOV    (SP)+,(R3)+      ;GET THE CONTENTS OF RH70//RM01 REG.
3571  014726  023705  014720                CMP    9$,R5            ;LAST REG. BEEN READ?
3572  014732  001414                        BEQ    12$              ;GET OUT IF YES
3573  014734  062737  000002  014720        ADD    #2,9$            ;INCREASE THE INDEX BY 2
3574  014742  000764                        BR     8$               ;LOOP--MORE TO READ
3575  014744  122703  000145          10$:  CMPB   #145,R3          ;IS IT A "SELECT DRIVE" COMMAND?
3576  014750  001405                        BEQ    12$              ;BRANCH IF YES
3577  014752  010346                  11$:  MOV    R3,-(SP)         ;LOAD THE COMMAND
3578  014754  004037  020032                JSR    R0,WRT.RM
3579  014760  000000                        RMCS1
3580  014762  015050                        CI7
3581  014764  004737  021102          12$:  JSR    PC,POPQUE        ;REMOVE REQ. FROM QUEUE
3582  014770  052762  000200  000016        BIS    #BIT07,16(R2)    ;SET THE "DONE" BIT
```

```
3583  014776  005737  012654          TST     SAVEFG          ;SAVE THE RH70/RM01 REGISTERS?
3584  015002  100002                  BPL     13$             ;BRANCH IF NO
3585  015004  004737  020224          JSR     PC,SVRH70       ;YES--GO SAVE THE REGISTERS
3586  015010  000207          13$:    RTS     PC              ;RETURN TO USER
3587  015012  006301          CI5:    ASL     R1
3588  015014  012761  001750  012660  MOV     #1000.,TIMER(R1)        ;SET A ONE SECOND TIMER
3589  015022  006201                  ASR     R1
3590  015024  112761  000001  012556  MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE
3591  015032  000207                  RTS     PC              ;RETURN TO THE USER
3592  015034  010346          CI6:    MOV     R3,-(SP)        ;LOAD THE COMMAND
3593  015036  004037  020032          JSR     R0,WRT.RM
3594  015042  000000                  RMCS1
3595  015044  015050                  CI7
3596  015046  000761                  BR      CI5
3597  015050  032764  010000  000010  CI7:    BIT     #BIT12,RMCS2(R4)  ;DRIVE NON-EXISTENT ?
3598  015056  001034                  BNE     CI8             ;BR IF YES
3599  015060  005702          1$:     TST     R2              ;ANYTHING IN QUEUE ?
3600  015062  001405                  BEQ     CI7B            ;BR IF NOT
3601  015064  012762  104000  000C16  MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
3602  015072  004737  020224          JSR     PC,SVRH70       ;GO SAVE THE RH70/RM01 REGISTERS
3603  015076  012746  000111  CI7B:   MOV     #111,-(SP)      ;DO A "DRIVE CLEAR"
3604  015102  004037  020032          JSR     R0,WRT.RM
3605  015106  000000                  RMCS1
3606  015110  015150                  CI8
3607  015112  004737  020764          JSR     PC,EMPTYQ       ;EMPTY THE QUEUE
3608  015116  105061  012634          CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD IN QUEUE FLAG
3609  015122  105061  012556          CLRB    DRVACT(R1)      ;DRIVE IS IDLE
3610  015126  020137  012700          CMP     R1,DTUW         ;IF THIS DRIVE HAD AN I/O REQUEST
3611  015132  001005                  BNE     1$              ;IN PROGRESS CLEAR ALL OF THE FLAGS
3612  015134  005037  012626          CLR     TRNSWT
3613  015140  012737  177777  012700  MOV     #-1,DTUW
3614  015146  000207          1$:     RTS     PC
3615  015150  104412          CI8:    SAVREG                  ;SAVE R0 - R5
3616  015152  032764  010000  000010  BIT     #BIT12,RMCS2(R4)  ;IS 'NED' SET ?
3617  015160  001002                  BNE     1$              ;BR IF YES
3618  015162  005001                  CLR     R1
3619  015164  005003                  CLR     R3
3620  015166  105761  012556          TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
3621  015172  001443                  BEQ     5$              ;BRANCH IF NO
3622  015174  013702  012626          MOV     TRNSWT,R2       ;GET THE "TRANSFER WAIT" QUEUE
3623  015200  020137  012700          CMP     R1,DTUW         ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
3624  015204  001402                  BEQ     2$              ;BRANCH IF YES
3625  015206  004737  021060          JSR     PC,GETREQ       ;GET THE DPB POINTER
3626  015212  005702          2$:     TST     R2              ;QUEUE ENTRY FOR DRIVE ?
3627  015214  001415                  BEQ     4$              ;BR IF NOT
3628  015216  032764  010000  000010  BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
3629  015224  001404                  BEQ     3$              ;BR IF NOT
3630  015226  012762  100002  000016  MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
3631  015234  000405                  BR      4$              ;CONTINUE
3632  015236  012762  102000  000016  3$:     MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
3633  015244  004737  020224          JSR     PC,SVRH70       ;SAVE RH70/RM01 REGISTERS
3634  015250  012763  177777  012660  4$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
3635  015256  105061  012556          CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
3636  015262  020137  012700          CMP     R1,DTUW         ;IS THIS DRIVE SETUP FOR A TRANSFER
3637  015266  001005                  BNE     5$              ;BR IF NOT
3638  015270  012737  177777  012700  MOV     #-1,DTUW        ;RESET THE INDICATOR
```

```
 3639  015276  005037  012626                  CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
 3640  015302  105061  012634          5$:     CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
 3641  015306  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
 3642  015314  001021                          BNE     6$              ;BR IF YES
 3643  015316  005201                          INC     R1              ;MOVE TO THE NEXT DRIVE
 3644  015320  062703  000002                  ADD     #2,R3
 3645  015324  042701  177770                  BIC     #^C7,R1
 3646  015330  001316                          BNE     1$              ;BRANCH IF MORE DRIVES
 3647  015332  012737  177777  012700          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
 3648  015340  005037  012626                  CLR     TRNSWT          ;CLEAR THE 'TRANSFER WAIT' QUEUE
 3649  015344  004737  020706                  JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
 3650  015350  012764  000040  000010          MOV     #BIT05,RMCS2(R4) ;DO A MASSBUS INIT.
 3651  015356  000406                          BR      7$              ;CONTINUE
 3652  015360  004737  020764          6$:     JSR     PC,EMPTYQ       ;CLEAR THE DRIVE'S QUEUE
 3653  015364  105061  012566                  CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
 3654  015370  105061  012576                  CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
 3655  015374  004737  020342          7$:     JSR     PC,SET.IE       ;SET "IE" WITHOUT "TRE"
 3656  015400  104413                          RESREG                  ;RESTORE R0 - R5
 3657  015402  000207                          RTS     PC              ;RETURN
 3658
 3659                                  ;LOOK AHEAD ROUTINE
 3660                                  ;
 3661                                  ;CALL
 3662                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
 3663                                  ;       MOV     #DPB,R2         ;POINT TO DPB
 3664                                  ;       JSR     R0,LA           ;GO CHECK THE WINDOW
 3665                                  ;       RETURN1                 ;ERROR RETURN
 3666                                  ;       RETURN2                 ;START A SEARCH
 3667                                  ;       RETURN3                 ;START A DATA TRANSFER
 3668
 3669  015404  013704  012714          LA:     MOV     RMADR,R4        ;GET RMCS1'S ADDRESS
 3670  015410  010164  000010                  MOV     R1,RMCS2,R4)    ;SELECT DRIVE
 3671  015414  004037  017652                  JSR     R0,RD.RM        ;READ DRIVE STATUS
 3672  015420  000012                          RMDS
 3673  015422  015552                          4$                      ;ERROR RETURN ADDRESS
 3674  015424  042716  157577                  BIC     #^C020200,(SP)  ;ON CYLINDER ?
 3675  015430  022726  000200                  CMP     #200,(SP)+      ;PIP=0,DRY=1?
 3676  015434  001044                          BNE     3$              ;NO
 3677  015436  105261  012644                  INCB    LACNT(R1)       ;INCREMENT THE LOOK AHEAD COUNT
 3678  015442  126137  012644  012722          CMPB    LACNT(R1),MXLACT ;EXCEED MAX?
 3679  015450  003033                          BGT     2$              ;BRANCH IF YES
 3680  015452  116203  000010                  MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
 3681  015456  000303                          SWAB    R3              ;MULT. BY 64--ALIGN WITH
 3682  015460  006203                          ASR     R3              ;LOOK AHEAD REGISTER
 3683  015462  006203                          ASR     R3
 3684  015464  012737  000340  177776          MOV     #340,@#PS       ;PRIORITY LEVEL "7"
 3685  015472  004037  017652          6$:     JSR     R0,RD.RM        ;READ LOOK AHEAD REGISTER
 3686  015476  000020                          RMLA
 3687  015500  015552                          4$
 3688  015502  021664  000020                  CMP(SP),RMLA(R4)        ;CORRECT LA NUMBER ?
 3689  015506  001402                          BEQ     7$              ;YES
 3690  015510  005726                          TST     (SP)+           ;NO,CLEAR STACK
 3691  015512  000415                          BR      3$
 3692  015514  162603                  7$:     SUB     (SP)+,R3        ;CALCULATE THE DELTA
 3693  015516  002002                          BGE     1$
 3694  015520  062703  004000                  ADD     #^D32.*64.,R3   ;MAKE THE DELTA POSITIVE
```

```
3695  015524  023703  012724          1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
3696  015530  002406                          BLT     3$              ;IF IT IS WITHIN THE
3697  015532  023703  012726                  CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
3698  015536  002003                          BGE     3$              ;THE LOOK AHEAD COUNT
3699  015540  105061  012644          2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
3700  015544  005720                          TST     (R0)+
3701  015546  005720                  3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
3702  015550  000402                          BR      5$              ;EXIT
3703  015552  004737  015050          4$:     JSR     PC,CI7          ;PROCESS THE ERROR
3704  015556  000200                  5$:     RTS     R0              ;RETURN
3705
3706                                  ;INTERRUPT SERVICE ROUTINE
3707
3708  015560  112737  000001  012632  ISR:    MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
3709  015566  104412                          SAVREG                  ;SAVE R0 - R5
3710  015570  013704  012714                  MOV     RMADR,R4        ;ADDRESS OF RHSCS1
3711  015574  013701  012700                  MOV     DTJW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
3712  015600  002403                          BLT     1$              ;BRANCH IF NO DATA TRANSFER UNDERWAY
3713  015602  004737  015624                  JSR     PC,TD           ;CALL TRANSFER DONE
3714  015606  000402                          BR      2$              ;EXIT
3715  015610  004737  015764          1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
3716  015614  104413                  2$:     RESREG                  ;RESTORE R0 - R5
3717  015616  105037  012632                  CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
3718  015622  000002                          RTI                     ;RETURN
3719
3720                                  ;TRANSFER DONE ROUTINE
3721
3722  015624  105061  012556          TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
3723  015630  012737  177777  012700          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
3724  015636  006301                          ASL     R1
3725  015640  012761  177777  012660          MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
3726  015646  006201                          ASR     R1
3727  015650  013702  012626                  MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
3728  015654  005037  012626                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
3729  015660  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET DONE
3730  015666  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
3731  015672  004037  017652                  JSR     R0,RD.RM        ;TRANSFER ERROR(TRE=1)?
3732  015676  000000                          RMCS1
3733  015700  015050                          CI7
3734  015702  006126                          ROL     (SP)+
3735  015704  100413                          BMI     3$              ;BR IF YES
3736  015706  005737  012654                  TST     SAVEFG          ;SAVE THE RH70/RM01 REGISTERS?
3737  015712  100002                          BPL     1$              ;BRANCH IF NO
3738  015714  004737  020224                  JSR     PC,SVRH70       ;YES--SAVE THE REGISTERS
3739  015720  004737  013730          1$:     JSR     PC,OPT          ;CALL OPTIMIZER
3740  015724  000417                          BR      SC              ;CHECK OTHER DRIVES
3741  015726  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
3742  015732  000414                          BR      SC              ;CHECK FOR OTHER DRIVES
3743  015734  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
3744  015742  004737  020764                  JSR     PC,EMPTYQ       ;EMPTY THE "DRIVE'S WAIT" QUEUE
3745  015746  004737  020224                  JSR     PC,SVRH70       ;SAVE THE RH70/RM01 REGISTERS
3746  015752  012714  040111                  MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
3747  015756  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
3748  015762  000400                          BR      SC              ;CHECK FOR OTHER DRIVES
3749
3750                                  ;SPECIAL CONDITION ROUTINE
```

```
3751
3752  015764  116403  000016          SC:     MOVB    RMAS(R4),R3     ;READ "RMAS"
3753  015770  001014                          BNE     2$              ;BRANCH IF ANY 'ATA' BITS SET
3754  015772  004037  017652                  JSR     R0,RD.RM        ;READ CONTROL AND STATUS REGISTER
3755  015776  000000                          RMCS1
3756  016000  015150                          CI8
3757  016002  106126                          ROLB    (SP)+           ;IS "IE"=1?
3758  016004  100405                          BMI     1$              ;YES, NO DRIVES TO CHECK
3759  016006  004037  021144                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
3760  016012  104001                          ERROR   1               ;REPORT AN ILLEGAL INTERRUPT
3761  016014  004737  020342                  JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
3762  016020  000207                  1$:     RTS     PC              ;RETURN
3763  016022  005046                  2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
3764  016024  110316                          MOVB    R3,(SP)         ;AN "ATA"=1
3765  016026  012703  000001                  MOV     #1,R3
3766  016032  005001                          CLR     R1
3767  016034  030316                  SC3:    BIT     R3,(SP)         ;ATA=1?
3768  016036  001005                          BNE     SC5             ;YES--BRANCH
3769  016040  005201                  SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
3770  016042  106303                          ASLB    R3
3771  016044  001373                          BNE     SC3             ;BRANCH IF MORE TO CHECK?
3772  016046  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
3773  016050  000207                          RTS     PC              ;RETURN TO USER
3774  016052  105761  012606          SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
3775  016056  001402                          BEQ     1$              ;BR IF NOT
3776  016060  000137  016764                  JMP     SC13            ;PROCESS THE DRIVE
3777  016064  105761  012616          1$:     TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
3778  016070  001402                          BEQ     2$              ;BR IF NOT
3779  016072  000137  016764                  JMP     SC13            ;START THE OUTSTANDING COMMAND
3780  016076  105761  012566          2$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
3781  016102  003025                          BGT     5$              ;BRANCH IF ONLINE
3782  016104  105761  012634                  TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
3783  016110  003422                          BLE     5$              ;BRANCH IF NOT
3784  016112  004737  021060                  JSR     PC,GETREQ       ;GET DPB POINTER
3785  016116  004737  020224                  JSR     PC,SVRH70       ;SAVE THE RH70/RM01 REGISTERS
3786  016122  004737  016714                  JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
3787                                                                  ;ALSO DO A DRIVE INIT (DRVINT)
3788  016126  105761  012566                  TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
3789  016132  003416                          BLE     6$              ;NO---BRANCH
3790  016134  032737  040000  012546          BIT     #BIT14,RMERRS   ;WAS THERE AN ERROR?
3791  016142  001002                          BNE     3$              ;BR IF ERROR
3792  016144  000137  016616                  JMP     SC11            ;NO ERROR
3793  016150  013705  012550          3$:     MOV     RMERRS+2,R5     ;YES -- PICKUP RMER1 AND
3794  016154  000502                          BR      SC6A            ;GO PROCESS THE ERROR
3795  016156  105761  012556          5$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
3796  016162  001033                          BNE     SC6             ;BR IF EITHER
3797  016164  004737  016714                  JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
3798                                                                  ;ALSO DO A DRVINT
3799  016170  105761  012606          6$:     TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
3800  016174  001321                          BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
3801  016176  105761  012566                  TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
3802  016202  100412                          BMI     7$              ;BR IF UNSAFE
3803  016204  032737  020000  012554          BIT     #BIT13,RMERRS+6 ;ADDRESS PLUG CHANGED ?
3804  016212  001013                          BNE     8$              ;BR IF YES
3805  016214  012746  000113                  MOV     #113,-(SP)      ;RELEASE COMMAND
3806  016220  004037  020032                  JSR     PC,WRT.RM       ;WRITE THE COMMAND INTO RMCS1
```

```
3807  016224  000000                          RMCS1                   ;REGISTER INDEX
3808  016226  016566                          SC8                     ;PARITY EXIT ADDRESS
3809  016230  011605                  7$:     MOV     (SP),R5         ;PICKUP (RMAS) BEFORE THE ERROR CALL
3810  016232  004037  021144                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
3811  016236  104002                          ERROR   2               ;REPORT THE UNEXPECTED ATTENTION
3812  016240  000677                          BR      SC4             ;GO CHECK FOR MORE ATA'S
3813  016242                          8$:
3814  016242  004037  021144                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
3815  016246  104005                          ERROR   5               ;REPORT THE ADDRESS PLUG CHANGE
3816  016250  000673                          BR      SC4             ;CHECK FOR MORE DRIVES
3817  016252  006301                  SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
3818  016254  012761  177777  012660          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
3819  016262  006201                          ASR     R1              ;RESTORE THE DRIVE ADDRESS
3820  016264  004737  021060                  JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
3821  016270  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
3822  016274  004037  017652                  JSR     R0,RD.RM        ;READ THE RM01'S STATUS REG.
3823  016300  000012                          RMDS
3824  016302  016566                          SC8
3825  016304  011605                          MOV     (SP),R5         ;AND PUT IT IN R5
3826  016306  006126                          ROL     (SP)+           ;WAS THERE AN ERROR?
3827  016310  100407                          BMI     1$              ;BR IF ERROR
3828  016312  105761  012556                  TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
3829  016316  003137                          BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
3830  016320  052762  100210  000016          BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
3831  016326  000470                          BR      SC7
3832  016330  004037  017652          1$:     JSR     R0,RD.RM        ;READ ERROR REGISTER #1
3833  016334  000014                          RMER1
3834  016336  016566                          SC8
3835  016340  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
3836  016342  004737  020224                  JSR     PC,SVRH70       ;SAVE RH70/RM01 REGISTERS
3837  016346  012746  000111                  MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
3838  016352  004037  020032                  JSR     R0,WRT.RM
3839  016356  000000                          RMCS1
3840  016360  016566                          SC8
3841  016362  006105                  SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
3842  016364  100406                          BMI     1$              ;BRANCH IF YES
3843  016366  005702                          TST     R2              ;ANYTHING IN QUEUE ?
3844  016370  001447                          BEQ     SC7             ;BR IF NOT
3845  016372  052762  100240  000016          BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
3846  016400  000443                          BR      SC7
3847  016402  004037  017652          1$:     JSR     R0,RD.RM        ;READ DRIVE STATUS REG. #1
3848  016406  000012                          RMDS
3849  016410  016566                          SC8
3850  016412  011605                          MOV     (SP),R5         ;SAVE RMDS IN R5
3851  016414  006126                          ROL     (SP)+           ;"ERR"=1?
3852  016416  100011                          BPL     2$              ;BR IF NO--UNSAFE CLEARED
3853  016420  112761  177777  012566          MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
3854  016426  004737  020224                  JSR     PC,SVRH70       ;SAVE RH70/RM01 REGISTERS
3855  016432  052762  110000  000016          BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
3856  016440  000423                          BR      SC7
3857  016442  032705  010000          2$:     BIT     #BIT12,R5       ;"MOL" = 1 ?
3858  016446  001015                          BNE     3$              ;BR IF YES
3859  016450  112761  177777  012556          MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
3860  016456  112761  000001  012566          MOVB    #1,DRVSTA(R1)   ;ONLINE
3861  016464  006301                          ASL     R1
3862  016466  012761  072460  012660          MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
```

```
3863  016474  006201                          ASR     R1
3864  016476  000137  016040                  JMP     SC4
3865  016502  052762  100220  000016  3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
3866  016510  105061  012556          SC7:    CLRB    DRVACT(R1)              ;DRIVE IS IDLE
3867  016514  004737  020764                  JSR     PC,EMPTYQ               ;DUMP THE QUEUE
3868  016520  105761  012634                  TSTB    ULDFLG(R1)              ;UNLOAD IN RMOGRESS OR QUELE?
3869  016524  003002                          BGT     1$                      ;BR IF NOT
3870  016526  105061  012634                  CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
3871  016532  116164  012702  000016  1$:     MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR ATTENTION BIT
3872  016540  105761  012566                  TSTB    DRVSTA(R1)              ;IS THE DRIVE UNSAFE ?
3873  016544  100406                          BMI     2$                      ;BR IF IT IS
3874  016546  012746  000113                  MOV     #113,-(SP)              ;RELEASE COMMAND
3875  016552  004037  020032                  JSR     R0,WRT.RM               ;WRITE THE COMMAND INTO RPCS1
3876  016556  000000                          RMCS1                           ;REGISTER INDEX
3877  016560  016566                          SC8                             ;PARITY EXIT ADDRESS
3878  016562  000137  016040          2$:     JMP     SC4                     ;CHECK FOR MORE DRIVES
3879  016566  105761  012556          SC8:    TSTB    DRVACT(R1)              ;IS DRIVE IDLE?
3880  016572  001405                          BEQ     1$                      ;YES--BRANCH
3881  016574  004737  021060                  JSR     PC,GETREQ               ;GET DPB POINTER
3882  016600  004737  015050                  JSR     PC,CI7                  ;PROCESS THE PARITY ERROR
3883  016604  000402                          BR      2$                      ;CONTINUE
3884  016606  004737  015076          1$:     JSR     PC,CI7B                 ;PROCESS THE UNCORRECTABLE PARITY ERROR
3885  016612  000137  016040          2$:     JMP     SC4                     ;CHECK MORE DRIVES
3886  016616  105761  012634          SC11:   TSTB    ULDFLG(R1)              ;"UNLOAD IN PROGRESS"?
3887  016622  003402                          BLE     1$                      ;BRANCH IF NO
3888  016624  105061  012634                  CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
3889  016630  105061  012556          1$:     CLRB    DRVACT(R1)              ;SET DRIVE IDLE
3890  016634  136137  012702  012630          BITB    ATABIT(R1),SRCHWT   ;DOING A SEARCH OPERATION FOR
3891                                                                          ;AN I/O COMMAND?
3892  016642  001012                          BNE     2$                      ;BRANCH IF YES
3893  016644  004737  021102                  JSR     PC,POPQUE               ;REMOVE REQUEST FROM QUELE
3894  016650  052762  000200  000016          BIS     #BIT07,16(R2)           ;SET "DONE" BIT
3895  016656  005737  012654                  TST     SAVEFG                  ;SAVE THE REGISTERS?
3896  016662  100002                          BPL     2$                      ;BRANCH IF NO
3897  016664  004737  020224                  JSR     PC,SVRH70               ;YES--SAVE ALL OF THE RH70/RM01 REGS
3898  016670  116164  012702  000016  2$:     MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR ATTENTION BIT
3899  016676  146137  012702  012630          BICB    ATABIT(R1),SRCHWT       ;CLEAR IMPLIED SEEK SET
3900  016704  004737  013730                  JSR     PC,OPT                  ;START A REQUEST
3901  016710  000137  016040                  JMP     SC4                     ;CHECK FOR MORE DRIVES
3902  016714  010164  000010          SC12:   MOV     R1,RMCS2(R4)            ;SELECT DRIVE
3903  016720  016437  000012  012546          MOV     RMDS(R4),RMERRS         ;SAVE THE FOUR REGISTERS THAT
3904  016726  016437  000014  012550          MOV     RMER1(R4),RMERRS+2      ;WILL TELL US SOMETHING
3905  016734  016437  000042  012552          MOV     RMER2(R4),RMERRS+4
3906  016742  016437  000040  012554          MOV     RMMR2(R4),RMERRS+6
3907  016750  004037  013144                  JSR     R0,DRVINT               ;INIT. THE STATE OF THE DRIVE
3908  016754  000401                          BR      1$                      ;TAKE ERROR EXIT
3909  016756  000207                          RTS     PC                      ;RETURN
3910  016760  005726                  1$:     TST     (SP)+                   ;POP PC OFF OF THE STACK
3911  016762  000701                          BR      SC8                     ;PROCESS THE PARITY ERROR
3912  016764  006301                  SC13:   ASL     R1                      ;SETUP TO ADDRESS WORDS
3913  016766  012761  177777  012660          MOV     #-1,TIMER(R1)           ;STOP THE TIMER
3914  016774  006201                          ASR     R1                      ;
3915  016776  010164  000010                  MOV     R1,RMCS2(R4)            ;SELECT THE DRIVE
3916  017002  116164  012702  000016          MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR THE ATTENTION BIT
3917  017010  032714  004000                  BIT     #BIT11,(R4)             ;DRIVE AVAILABLE ?
3918  017014  001006                          BNE     1$                      ;BR IF AVAILABLE
```

```
3919  017016  006301                          ASL     R1
3920  017020  012761  023420  012660          MOV     #10000.,TIMER(R1)  ;START 10 SEC TIMER AGAIN
3921  017026  006201                          ASR     R1
3922  017030  000433                          BR      3$                 ;EXIT
3923  017032  105761  012606          1$:     TSTB    DPINT(R1)          ;INITIALIZING THE DRIVE ?
3924  017036  001424                          BEQ     2$                 ;BR IF NOT
3925  017040  105061  012606                  CLRB    DPINT(R1)          ;CLEAR THE INIT INDICATOR
3926  017044  004037  013144                  JSR     R0,DRVINT          ;GO INIT THE DRIVE
3927  017050  000240                          NOP                        ;DUMMY PARITY ERROR RETURN
3928  017052  105761  012566                  TSTB    DRVSTA(R1)         ;DRIVE ONLINE ?
3929  017056  003014                          BGT     2$                 ;BR IF YES -- START ORDER
3930  017060  005702                          TST     R2                 ;QUEUE ENTRY FOR THE DRIVE
3931  017062  001416                          BEQ     3$                 ;BR IF NOT
3932  017064  004737  021060                  JSR     PC,GETREQ          ;GET DPB ADDRESS
3933  017070  052762  140000  000016          BIS     #BIT15!BIT14,16.(R2)  ;INFORM USER THAT DRIVE OFFLINE
3934  017076  004737  020224                  JSR     PC,SVRH70          ;SAVE THE REGISTERS
3935  017102  004737  020764                  JSR     PC,EMPTYQ          ;EMPTY THE REQUEST QUEUE
3936  017106  000404                          BR      3$
3937  017110  105061  012616          2$:     CLRB    DPRQS(R1)          ;CLEAR THE PORT REQUEST INDICATOR
3938  017114  004737  013730                  JSR     PC,OPT             ;START THE PENDING REQUEST
3939  017120  000137  016040          3$:     JMP     SC4                ;PROCESS OTHER DRIVES
3940
3941                                  ;/RM01 TIMER ROUTINE
3942                                  ;CALL
3943                                  ;       MOV     #TIME,-(SP)        ;ELAPSED TIME IN MILLISECONDS ON THE STACK
3944                                  ;       JSR     PC,RPTMR           ;CALL RM01 TIME ROUTINE
3945
3946  017124  005737  012632          RPTMR:  TST     ACTDRV             ;CHECK "ACTDRV & ACTSTR"
3947  017130  001030                          BNE     4$                 ;IF NON ZERO EXIT
3948  017132  112737  000001  012633          MOVB    #1,ACTSTR          ;SET "ACTSTR"
3949  017140  104412                          SAVREG                     ;SAVE R0 - R5
3950  017142  005001                          CLR     R1                 ;START WITH DRIVE 0
3951  017144  005003                          CLR     R3
3952  017146  005763  012660          1$:     TST     TIMER(R3)          ;IS THE TIMER RUNNING?
3953  017152  002407                          BLT     2$                 ;BRANCH IF NO
3954  017154  166663  000002  012660          SUB     2(SP),TIMER(R3)    ;COUNT THE INTERVAL
3955  017162  003003                          BGT     2$                 ;BR IF NO SOFTWARE TIMEOUT
3956  017164  004737  017216                  JSR     PC,STO             ;CALL SOFTWARE TIMEOUT ROUTINE
3957  017170  000405                          BR      3$                 ;GO TO THE EXIT
3958  017172  005201                  2$:     INC     R1                 ;MOVE TO NEXT DRIVE
3959  017174  005723                          TST     (R3)+
3960  017176  022701  000010                  CMP     #8.,R1             ;OUT OF DRIVES?
3961  017202  003361                          BGT     1$                 ;BRANCH IF NO
3962  017204  104413                  3$:     RESREG                     ;RESTORE R0 - R5
3963  017206  105037  012633                  CLRB    ACTSTR             ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
3964  017212  012616                  4$:     MOV     (SP)+,(SP)         ;ADJUST THE STACK
3965  017214  000207                          RTS     PC                 ;RETURN
3966
3967                                  ;SOFTWARE TIMEOUT ROUTINE
3968                                  ;
3969                                  ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
3970                                  ;       OR GREATER
3971                                  ;
3972                                  ;CALL:  STO
3973                                  ;       MOV     #DRVNUM,R1         ;DRIVE NUMBER
3974                                  ;       JSR     PC,STO             ;CALL
```

```
3975                                        ;       RETURN
3976
3977  017216  010146                        STO:    MOV     R1,-(SP)        ;SAVE R1
3978  017220  010346                                MOV     R3,-(SP)        ;SAVE R3
3979  017222  013704  012714                        MOV     RMADR,R4        ;GET ADDRESS OF "RMCS1"
3980  017226  010164  000010                        MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
3981  017232  004037  017652                        JSR     R0,RD.RM        ;READ "DRIVE STATUS REG"
3982  017236  000012                                RMDS
3983  017240  017540                                STOS
3984  017242  105726                                TSTB    (SP)+           ;IS "DRY"=1?
3985  017244  100477                                BMI     STO2            ;BR IF YES
3986  017246  105761  012606              STO1:     TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
3987  017252  001074                                BNE     STO2            ;BR IF YES
3988  017254  105761  012616                        TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
3989  017260  001071                                BNE     STO2            ;BR IF YES
3990  017262  013702  012626                        MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
3991  017266  020137  012700                        CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
3992  017272  001402                                BEQ     1$              ;BRANCH IF YES
3993  017274  004737  021060                        JSR     PC,GETREQ       ;GET DPB ADDRESS
3994  017300  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
3995  017306  004737  020224                        JSR     PC,SVRH70       ;SAVE RH70/RM01 REGISTERS
3996  017312  012764  000040  000010                MOV     #BIT05,RMCS2(R4) ;"INIT" THE MASS BUS
3997  017320  105061  012556                        CLRB    DRVACT(R1)      ;DRIVE IS IDLE
3998  017324  105061  012634                        CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
3999  017330  005001                                CLR     R1              ;START WITH DRIVE 0
4000  017332  005003                                CLR     R3
4001  017334  004037  013144              2$:     JSR     R0,DRVINT       ;INIT. THIS DRIVE
4002  017340  000477                                BR      STO5            ;PARITY ERROR RETURN
4003  017342  105761  012556                        TSTB    DRVACT(R1)      ;DRIVE IDLE BEFORE THE INIT.?
4004  017346  001414                                BEQ     4$              ;YES--BRANCH
4005  017350  013702  012626                        MOV     TRNSWT,R2       ;GET TRANSFER WAIT QUEJE
4006  017354  023701  012700                        CMP     DTUW,R1         ;WAS THERE I/O ON THIS DRIVE?
4007  017360  001402                                BEQ     3$              ;YES--BRANCH
4008  017362  004737  021060                        JSR     PC,GETREQ       ;GET THE DPB POINTER FROM QUEUE
4009  017366  052762  100400  000016  3$:     BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT.
4010  017374  105061  012556                        CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE TO IDLE
4011  017400  105061  012634              4$:     CLRB    ULDFLG(R1)      ;NO UNLOAD
4012  017404  012763  177777  012660                MOV     #-1,TIMER(R3)   ;STOP THE TIMER
4013  017412  005723                                TST     (R3)+           ;UPDATE THE INDEX
4014  017414  005201                                INC     R1              ;INCREMENT THE DRIVE NUMBER
4015  017416  022701  000010                        CMP     #8.,R1          ;LAST DRIVE BEEN CHECKED?
4016  017422  003344                                BGT     2$              ;NO--LOOP
4017  017424  012737  177777  012700                MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
4018  017432  005037  012626                        CLR     TRNSWT          ;CLEAR TRANSFER WAIT QUELE
4019  017436  004737  020706                        JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
4020  017442  000500                                BR      STO9            ;EXIT
4021  017444  116405  000016              STO2:     MOVB    RMAS(R4),R5     ;READ ATTENTION REG
4022  017450  136105  012702                        BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
4023  017454  001017                                BNE     STO3            ;YES--BRANCH
4024  017456  105761  012606                        TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
4025  017462  001031                                BNE     STO6            ;BR IF YES - DRIVE NOT ONLINE
4026  017464  105761  012616                        TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
4027  017470  001045                                BNE     STO7            ;BR IF YES - NO RESPONSE TO REQUEST
4028  017472  020137  012700                        CMP     R1,DTUW         ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
4029  017476  001263                                BNE     STO1            ;BR IF NO
4030  017500  004037  017652                        JSR     R0,RD.RM        ;YES--CHECK "RDY"
```

```
4031  017504  000000                          RMCS1
4032  017506  017540                          ST05
4033  017510  105726                          TSTB    (SP)+
4034  017512  100255                          BPL     ST01            ;BR IF "RDY"=0
4035  017514  105761  012606          ST03:   TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
4036  017520  001003                          BNE     1$              ;BR IF INIT PENDING
4037  017522  105761  012616                  TSTB    DPRQS(R1)       ;PORT REQUEST PENDING ?
4038  017526  001446                          BEQ     ST09            ;BR IF NOT
4039  017530  012763  177777  012660  1$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
4040  017536  000442                          BR      ST09            ;EXIT
4041  017540  004737  015150          ST05:   JSR     PC,CI8          ;GO HANDLE THE PARITY ERROR
4042  017544  000437                          BR      ST09
4043  017546  105061  012606          ST06:   CLRB    DPINT(R1)       ;CLEAR THE INITIALIZE INDICATOR
4044  017552  105061  012566                  CLRB    DRVSTA(R1)      ;SET UNIT OFFLINE
4045  017556  012763  177777  012660          MOV     #-1,TIMER(R3)   ;STOP THE TIMER
4046  017564  004737  021060                  JSR     PC,GETREQ       ;GET THE DPB ADDRESS
4047  017570  005702                          TST     R2              ;REQUEST IN QUEUE ?
4048  017572  001424                          BEQ     ST09            ;BR IF NOT
4049  017574  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM THE USER DRIVE NOT AVAILABLE
4050  017602  000414                          BR      ST08            ;FINISH
4051  017604  012763  177777  012660  ST07:   MOV     #-1,TIMER(R3)   ;STOP THE TIMER
4052  017612  105061  012616                  CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST INDICATOR
4053  017616  004737  021060                  JSR     PC,GETREQ       ;GET DPB ADDRESS
4054  017622  005702                          TST     R2              ;QUEUE ENTRY FOR DRIVE ?
4055  017624  001407                          BEQ     ST09            ;BR IF NONE
4056  017626  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)  ;INFORM USER OF PORT REQUEST ERROR
4057  017634  004737  020764          ST08:   JSR     PC,EMPTYQ       ;CLEAR THE QUEUE FOR THE DRIVE
4058  017640  004737  020224                  JSR     PC,SVRH70       ;SAVE THE REGISTERS
4059  017644  012603                  ST09:   MOV     (SP)+,R3        ;RESTORE R3
4060  017646  012601                          MOV     (SP)+,R1        ;RESTORE R1
4061  017650  000207                          RTS     PC              ;RETURN
4062
4063                                  ;ROUTINE TO READ A RH70/RM01 REGISTER
4064                                  ;
4065                                  ;CALL
4066                                  ;       JSR     R0,RD.RM        ;GO READ A REGISTER
4067                                  ;       INDEX                   ;REG. INDEX FROM BASE
4068                                  ;       ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
4069                                  ;                               ;AT THIS ADDRESS
4070                                  ;       RETURN                  ;CONTENTS OF REG. IS ON THE STACK
4071                                  ;
4072  017652  013737  012712  020020  RD.RM:   MOV     MCPEMX,RD.RM2   ;MAX. RETRYS ALLOWED
4073  017660  011646                          MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
4074  017662  013737  012714  017676          MOV     RMADR,RD.ADR    ;FORM THE DESIRED ADDRESS
4075  017670  062037  017676                  ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
4076  017674  013727                  RD.RM1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RM01
4077  017676  000000                  RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
4078  017700  000000                  RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
4079  017702  013766  017700  000002          MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
4080  017710  013746  012714                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
4081  017714  062716  000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
4082  017720  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
4083  017724  001037                          BNE     RD.RM3          ;BR IF DRIVE NON-EXISTENT
4084  017726  017746  172762                  MOV     @RMADR,-(SP)    ;READ RMCS1
4085  017732  032716  020000                  BIT     #BIT13,(SP)     ;DID MCPE SET?
4086  017736  001002                          BNE     1$              ;BRANCH IF YES
```

```
4087  017740  022620                        CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
4088  017742  000432                        BR      RD.RM4          ;EXIT
4089  017744                        1$:
4090  017744  004037  021144                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
4091  017750  104003                        ERROR   3               ;REPORT "MCPE" ERROR
4092  017752  005737  012700                TST     DTUW            ;DATA TRANSFER UNDERWAY?
4093  017756  100405                        BMI     2$              ;NO--BRANCH
4094  017760  032716  040000                BIT     #BIT14,(SP)     ;NO--"TRE"=1?
4095  017764  001402                        BEQ     2$              ;NO--BRANCH
4096  017766  005726                        TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
4097  017770  000415                        BR      RD.RM3          ;TAKE THE FATAL ERROR EXIT
4098  017772  052716  040000        2$:     BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
4099  017776  000316                        SWAB    (SP)            ;POSITION BEFORE WRITING
4100  020000  013737  012714  020014        MOV     RMADR,3$        ;FORM ADDRESS OF HIGH BYTE
4101  020006  005237  020014                INC     3$
4102  020012  112637                        MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RMCS1
4103  020014  000000        3$:     .WORD   0                       ;ADDRESS STORAGE
4104  020016  005327                        DEC     (PC)+           ;EXCEEDED MAX. RETRYS
4105  020020  000003        RD.RM2: .WORD   3
4106  020022  002324                        BGE     RD.RM1          ;BRANCH IF NO
4107  020024  011000        RD.RM3: MOV     (R0),R0                 ;FATAL ERROR EXIT
4108  020026  012616                        MOV     (SP)+,(SP)
4109  020030  000200        RD.RM4: RTS     R0
4110
4111                                ;ROUTINE TO WRITE A REGISTER
4112                                ;
4113                                ;CALL
4114                                ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
4115                                ;       JSR     R0,WRT.RM       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
4116                                ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
4117                                ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
4118                                ;       RETURN                  ;ERROR FREE RETURN
4119
4120  020032  013737  012712  020206  WRT.RM: MOV     MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
4121  020040  016637  000002  020120          MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
4122  020046  012616                        MOV     (SP)+,(SP)      ;ADJUST THE STACK
4123  020050  012037  020122                MOV     (R0)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
4124  020054  001015                        BNE     1$              ;BRANCH IF NOT RMCS1
4125  020056  122737  000150  020120        CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
4126  020064  002411                        BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
4127  020066  004037  017652                JSR     R0,RD.RM        ;NO---COMBINE A16&A17, & PSEL WITH
4128  020072  000000                        RMCS1                   ;THE COMMAND BEFORE SENDING IT TO
4129  020074  020214                        WRT.R3                  ;THE RH70/RM01
4130  020076  000316                        SWAB    (SP)
4131  020100  042716  177770                BIC     #^C7,(SP)
4132  020104  112637  020121                MOVB    (SP)+,WRT.WD+1
4133  020110  063737  012714  020122  1$:   ADD     RMADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
4134  020116  012737          WRT.R1: MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
4135  020120  000000          WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
4136  020122  000000          WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
4137  020124  013746  012714                MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
4138  020130  062716  000010                ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
4139  020134  032736  010000                BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
4140  020140  001025                        BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
4141  020142  004037  017652                JSR     R0,RD.RM        ;CHECK FOR PARITY ERROR ON WRITE
4142  020146  000014                        RMER1
```

```
 4143  020150  020214                          WRT.R3
 4144  020152  032726  000010                  BIT     #BIT03,(SP)+
 4145  020156  001420                          BEQ     WRT.R4          ;BRANCH IF "PAR=0"
 4146  020160  016037  177776  020172          MOV     -2(R0),1$       ;PICKUP THE INDEX
 4147  020166  004037  017552                  JSR     R0,RD.RM        ;READ THE REG.
 4148  020172  000000                  1$:     .WORD   0               ;REG. INDEX
 4149  020174  020214                          WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
 4150  020176  004037  021144                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
 4151  020202  104004                          ERROR   4               ;REPORT THE PARITY ON WRITE ERROR
 4152  020204  005327                          DEC     (PC)+           ;DECREMENT THE ERROR COUNT
 4153  020206  000003                  WRT.R2: .WORD   3               ;RETRY COUNTER
 4154  020210  002342                          BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
 4155  020212  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
 4156  020214  011000                  WRT.R3: MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
 4157  020216  000401                          BR      WRT.R5          ;EXIT
 4158  020220  005720                  WRT.R4: TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
 4159  020222  000200                  WRT.R5: RTS     R0
 4160
 4161                                  ;ROUTINE TO SAVE THE RH70/RP04/5/RM01 REGISTERS AS PER DPB+14
 4162                                  ;
 4163                                  ;CALL
 4164                                  ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
 4165                                  ;       JSR     PC,SVRH70       ;SAVE THE DRIVES REG'S
 4166
 4167  020224  104412                  SVRH70: SAVREG                  ;SAVE R0 - R5
 4168  020226  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
 4169  020230  001430                          BEQ     4$              ;BR IF NONE
 4170  020232  013704  012714                  MOV     RMADR,R4
 4171  020236  111264  000010                  MOVB    (R2),RMCS2(R4)  ;SELECT DRIVE
 4172  020242  016203  000014                  MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
 4173  020246  001433                          BEQ     6$              ;EXIT IF NO ADDRESS
 4174  020250  005037  020304                  CLR     3$              ;COUNTER & POINTER
 4175  020254  023727  020304  000022  1$:     CMP     3$,#RMDB        ;REACHED THE BUFFER REGISTER ?
 4176  020262  001006                          BNE     2$              ;BR IF NOT
 4177  020264  032764  000200  000010          BIT     #BIT07,RMCS2(R4)  ;'OR' SET ?
 4178  020272  001002                          BNE     2$              ;BR IF SET
 4179  020274  005023                          CLR     (R3)+           ;STORE RMDB AS ZEROES
 4180  020276  000405                          BR      4$              ;CONTINUE
 4181  020300  004037  017652          2$:     JSR     R0,RD.RM        ;READ THE SELECTED REGISTER
 4182  020304  000000                  3$:     .WORD   0               ;REGISTER INDEX
 4183  020306  020332                          5$                      ;ERROR RETURN ADDRESS
 4184  020310  012623                          MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
 4185  020312  023727  020304  000046  4$:     CMP     3$,#RMEC2       ;REACHED THE END ?
 4186  020320  001406                          BEQ     6$              ;BR IF YES
 4187  020322  062737  000002  020304          ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
 4188  020330  000751                          BR      1$              ;CONTINUE READING THE REGISTERS
 4189  020332  004737  015050          5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
 4190  020336  104413                  6$:     RESREG                  ;RESTORE R0 - R5
 4191  020340  000207                          RTS     PC              ;RETURN
 4192
 4193                                  ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
 4194                                  ;CALL
 4195                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 4196                                  ;       JSR     PC,SET.IE       ;SET "IE"
 4197                                  ;       RETURN
 4198
```

```
4199  020342  010446                    SET.IE: MOV     R4,-(SP)          ;SAVE R4
4200  020344  013704  012714                    MOV     RMADR,R4          ;PICKUP ADDRESS OF RMCS1
4201  020350  010164  000010                    MOV     R1,RMCS2(R4)      ;SELECT DRIVE
4202  020354  011446                            MOV     (R4),-(SP)        ;READ RMCS1
4203  020356  052716  040000                    BIS     #BIT14,(SP)       ;SET THE "TRE" BIT OF THE WORD READ
4204  020362  000316                            SWAB    (SP)              ;ADJUST FOR DATO
4205  020364  112714  000100                    MOVB    #BIT06,(R4)       ;SET "IE"
4206  020370  032764  010000  000010            BIT     #BIT12,RMCS2(R4)  ;IS "NED"=1?
4207  020376  001002                            BNE     1$                ;YES--CLEAR "TRE"
4208  020400  005726                            TST     (SP)+             ;CLEAN OFF THE STACK
4209  020402  000402                            BR      2$
4210  020404  112664  000001            1$:     MOVB    (SP)+,1(R4)       ;CLEAR "TRE"
4211  020410  012604                    2$:     MOV     (SP)+,R4          ;RESTORE R4
4212  020412  000207                            RTS     PC                ;RETURN TO CALLER
4213
4214                                    ;QUEUE COUNT
4215  020414      000                   QCNT:   .BYTE   0                 ;DRIVE 0
4216  020415      000                           .BYTE   0                 ;DRIVE 1
4217  020416      000                           .BYTE   0                 ;DRIVE 2
4218  020417      000                           .BYTE   0                 ;DRIVE 3
4219  020420      000                           .BYTE   0                 ;DRIVE 4
4220  020421      000                           .BYTE   0                 ;DRIVE 5
4221  020422      000                           .BYTE   0                 ;DRIVE 6
4222  020423      000                           .BYTE   0                 ;DRIVE 7
4223
4224                                    ;QUEUE INPUT POINTERS
4225
4226  020424  020506                    QINPT:  .WORD   QDRV0              ;DRIVE 0
4227  020426  020526                            .WORD   QDRV1             ;DRIVE 1
4228  020430  020546                            .WORD   QDRV2             ;DRIVE 2
4229  020432  020566                            .WORD   QDRV3             ;DRIVE 3
4230  020434  020606                            .WORD   QDRV4             ;DRIVE 4
4231  020436  020626                            .WORD   QDRV5             ;DRIVE 5
4232  020440  020646                            .WORD   QDRV6             ;DRIVE 6
4233  020442  020666                            .WORD   QDRV7             ;DRIVE 7
4234
4235                                    ;QUEUE OUTPUT POINTERS
4236
4237  020444  020506                    QOUTPT: .WORD   QDRV0              ;DRIVE 0
4238  020446  020526                            .WORD   QDRV1             ;DRIVE 1
4239  020450  020546                            .WORD   QDRV2             ;DRIVE 2
4240  020452  020566                            .WORD   QDRV3             ;DRIVE 3
4241  020454  020606                            .WORD   QDRV4             ;DRIVE 4
4242  020456  020626                            .WORD   QDRV5             ;DRIVE 5
4243  020460  020646                            .WORD   QDRV6             ;DRIVE 6
4244  020462  020666                            .WORD   QDRV7             ;DRIVE 7
4245
4246  020464  020506                    QSTART: .WORD   QDRV0              ;DRIVE 0 START ADDRESS
4247  020466  020526                    QSTOP:  .WORD   QDRV1              ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
4248  020470  020546                            .WORD   QDRV2             ;STOP DRIVE 1--START DRIVE 2
4249  020472  020566                            .WORD   QDRV3             ;STOP DRIVE 2--START DRIVE 3
4250  020474  020606                            .WORD   QDRV4             ;STOP DRIVE 3--START DRIVE 4
4251  020476  020626                            .WORD   QDRV5             ;STOP DRIVE 4--START DRIVE 5
4252  020500  020646                            .WORD   QDRV6             ;STOP DRIVE 5--START DRIVE 6
4253  020502  020666                            .WORD   QDRV7             ;STOP DRIVE 6--START DRIVE 7
4254  020504  020706                            .WORD   QTERM             ;STOP DRIVE 7
```

```
                                        ;DRIVE REQUEST QUEUES
                                        ;
020506  000010                          QDRV0:  .BLKW   10
020526  000010                          QDRV1:  .BLKW   10
020546  000010                          QDRV2:  .BLKW   10
020566  000010                          QDRV3:  .BLKW   10
020606  000010                          QDRV4:  .BLKW   10
020626  000010                          QDRV5:  .BLKW   10
020646  000010                          QDRV6:  .BLKW   10
020666  000010                          QDRV7:  .BLKW   10
        020706                          QTERM=.
                                        ;
                                        ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEJES
                                        ;
                                        ;CALL
                                        ;       JSR     PC,CLRQUE
                                        ;
020706  104412                          CLRQUE: SAVREG                  ;SAVE R0 - R5
020710  012702  020414                          MOV     #QCNT,R2        ;ZERO THE QUEJE COUNTS
020714  005022                                  CLR     (R2)+           ;DRIVES 0 & 1
020716  005022                                  CLR     (R2)+           ;DRIVES 2 & 3
020720  005022                                  CLR     (R2)+           ;DRIVES 4 & 5
020722  005022                                  CLR     (R2)+           ;DRIVES 6 & 7
020724  012703  000010                          MOV     #8.,R3          ;MOVE THE STARTING
020730  012701  020464                          MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
020734  012122                          1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPJT POINTER
020736  005303                                  DEC     R3
020740  001375                                  BNE     1$
020742  012703  000010                          MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
020746  012701  020464                          MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
020752  012122                          2$:     MOV     (R1)+,(R2)+     ;QUEUE OJTPUT POINTER
020754  005303                                  DEC     R3
020756  001375                                  BNE     2$
020760  104413                                  RESREG                  ;RESTORE R0 - R5
020762  000207                                  RTS     PC
                                        ;
                                        ;EMPTY THE QUEUE SPECIFIED BY R1
                                        ;
                                        ;CALL
                                        ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
                                        ;       JSR     PC,EMPTYQ
                                        ;
020764  105061  020414                  EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
020770  006301                                  ASL     R1
020772  016161  020424  020444                  MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
021000  006201                                  ASR     R1
021002  000207                                  RTS     PC
                                        ;
                                        ;ROUTINE TO PUT A REQJEST IN QUEUE
                                        ;
                                        ;CALL
                                        ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                        ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
                                        ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
                                        ;       RETURN1                 ;RETURN HERE IF QUEJE IS FULL
```

```
                                        ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
                                        ;
021004  122761  000010  020414  DRVQUE: CMPB    #10,QCNT(R1)            ;IS QUEUE FULL?
021012  001421                          BEQ     2$                      ;BR IF YES-TAKE RETURN1
021014  105261  020414                  INCB    QCNT(R1)                ;INCREMENT QUEUE COUNT
021020  006301                          ASL     R1
021022  010271  020424                  MOV     R2,@QINPT(R1)           ;PUT THIS REQUEST IN QUEUE
021026  062761  000002  020424          ADD     #2,QINPT(R1)            ;UPDATE THE QUEUE POINTER
021034  026161  020424  020466          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
021042  001003                          BNE     1$                      ;BRANCH IF NO
021044  016161  020464  020424          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
021052  006201                  1$:     ASR     R1
021054  005720                          TST     (R0)+                   ;TAKE RETURN 2
021056  000200                  2$:     RTS     R0                      ;RETURN TO USER
                                        ;
                                        ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
                                        ;
                                        ;CALL
                                        ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                        ;       JSR     PC,GETREQ       ;GO GET THE REQUEST
                                        ;       RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
                                        ;                               ;R2=0 IF NO REQUEST IN QUEUE
                                        ;
021060  005002                  GETREQ: CLR     R2
021062  105761  020414                  TSTB    QCNT(R1)                ;IS THERE ANY REQUEST IN QUEUE?
021066  001404                          BEQ     2$                      ;NO---BRANCH
021070  006301                  1$:     ASL     R1
021072  017102  020444                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
021076  006201                          ASR     R1
021100  000207                  2$:     RTS     PC                      ;RETURN TO USER
                                        ;
                                        ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
                                        ;
                                        ;CALL
                                        ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                        ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
                                        ;       RETURN                  ;R2=ADDRESS OF DPB REMOVED
                                        ;
021102  105361  020414          POPQUE: DECB    QCNT(R1)                ;DECREMENT QUEUE COUNT
021106  006301                          ASL     R1
021110  017102  020444                  MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
021114  062761  000002  020444          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
021122  026161  020444  020466          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
021130  001003                          BNE     1$                      ;NO--BRANCH TO EXIT
021132  016161  020464  020444          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
021140  006201                  1$:     ASR     R1
021142  000207                          RTS     PC                      ;RETURN TO USER
                                        ;
                                        ;ROUTINE TO SAVE THE CONTENTS OF '$ESCAPE' WHEN THE DRIVER
                                        ;REPORTS AN ERROR DIRECTLY.
                                        ;
                                        ;CALL
                                        ;       JSR     R0,ES.SAV
                                        ;       ERROR   N               ;;THE ERROR CALL
                                        ;       RETURN                  ;THE RETURN IS PAST THE ERROR CALL
```

```
 4367  021144  012037  021160          ES.SAV: MOV     (R0)+,1$        ;GET THE ERROR CALL
 4368  021150  013746  001160                  MOV     $ESCAPE,-(SP)   ;SAVE THE ADDRESS IN '$ESCAPE'
 4369  021154  005037  001160                  CLR     $ESCAPE         ;CLEAR THE ESCAPE RETURN
 4370  021160  000000                  1$:     .WORD   0               ;THE ERROR CALL IS MOVED HERE
 4371  021162  012637  001160                  MOV     (SP)+,$ESCAPE   ;RESTORE THE ESCAPE ADDRESS
 4372  021166  000200                          RTS     R0              ;RETURN
 4373
 4374
 4375                                          ;;*****************************************************************
 4376
 4377                                          .SBTTL  TELETYPE MESSAGES
 4378
 4379  021170  047440  043136  044514  UNTOFF: .ASCIZ  / OFFLINE/
 4380  021176  042516     000
 4381  021201     040  047117  044514  UNTON:  .ASCIZ  / ONLINE/
 4382  021206  042516     000
 4383  021211     040  047516  020124  NOTRM:  .ASCIZ  @ NOT AN RM01@
 4384  021216  047101  051040  030115
 4385  021224  000061
 4386  021226  047040  052117  050040  NOTPRS: .ASCIZ  / NOT PRESENT/
 4387  021234  042522  042523  052116
 4388  021242     000
 4389  021243     040  047125  040523  NOTSAF: .ASCIZ  / UNSAFE/
 4390  021250  042506     000
 4391  021253     122  030120  000064  RM01B:  .ASCIZ  /RP04/
 4392  021260  050122  032460     000  RP05:   .ASCIZ  /RP05/
 4393  021265     122  030115  000061  RM01A:  .ASCIZ  /RM01/
 4394  021272  047125  052111  051440  SYSTAT: .ASCIZ  /UNIT STATUS/<CR><LF><LF>
 4395  021300  040524  052524  006523
 4396  021306  005012     000
 4397  021311     015  042012  044522  MUNIT:  .ASCIZ  <CR><LF>/DRIVE: /
 4398  021316  042526  020072     000
 4399  021323     040  020057     000  SLASH:  .ASCIZ  @ / @
 4400  021327     103     000          C:      .ASCIZ  /C/
 4401  021331     124     000          T:      .ASCIZ  /T/
 4402  021333     060     000          MDRVD:  .ASCIZ  /0/
 4403  021335     040     040          LIN4SP: .ASCII  /  /
 4404  021337     040  000040          LINSP:  .ASCIZ  / /
 4405  021342  047105  042524  020122  ENTADR: .ASCIZ  /ENTER ADDRESS LIMITS:/<CR><LF>
 4406  021350  042101  051104  051505
 4407  021356  020123  044514  044515
 4408  021364  051524  006472  000012
 4409  021372  020040  051104  053111  MOFF_N: .ASCIZ  /  DRIVE OFFLINE/<CR><LF>
 4410  021400  020105  043117  046106
 4411  021406  047111  006505  000012
 4412  021414  042040  044522  042526  MDRNP:  .ASCIZ  / DRIVE NOT PRESENT/<CR><LF>
 4413  021422  047040  052117  050040
 4414  021430  042522  042523  052116
 4415  021436  005015     000
 4416  021441     040  051104  053111  MER11:  .ASCIZ  / DRIVE NOT AVAILABLE/<CR><LF>
 4417  021446  020105  047516  020124
 4418  021454  053101  044501  040514
 4419  021462  046102  006505  000012
 4420  021470  042040  044522  042526  MNRM01: .ASCIZ  @ DRIVE NOT AN RM01@<CR><LF>
 4421  021476  047040  052117  040440
 4422  021504  020116  046522  030460
```

```
 4423  021512  005015     000
 4424  021515     040  051104  053111  MUSDR:  .ASCIZ  / DRIVE UNSAFE/<CR><LF>
 4425  021522  020105  047125  040523
 4426  021530  042506  005015     000
 4427  021535     040  042523  042514  MSELD:  .ASCIZ  / SELECTED/
 4428  021542  052103  042105     000
 4429  021547     015  050012  047522  MMODE:  .ASCIZ  <CR><LF>/PROGRAM MODE (C OR F): /
 4430  021554  051107  046501  046440
 4431  021562  042117  020105  041450
 4432  021570  047440  020122  024506
 4433  021576  020072     000
 4434  021601     040  047506  046522  MFORMT: .ASCIZ  / FORMAT & VERIFY/
 4435  021606  052101  023040  053040
 4436  021614  051105  043111  000131
 4437  021622  044103  041505  020113  MHECK:  .ASCIZ  /CHECK ONLY/
 4438  021630  047117  054514     000
 4439  021635     106  051117  040515  MORMAT: .ASCIZ  /FORMAT & VERIFY/
 4440  021642  020124  020046  042526
 4441  021650  044522  054506     000
 4442  021655     015  005012  050117  MSIZE:  .ASCIZ  <CR><LF><LF>/OPERATE IN 32 SECTOR MODE (Y OR N) ? /
 4443  021662  051105  052101  020105
 4444  021670  047111  031440  020062
 4445  021676  042523  052103  051117
 4446  021704  046440  042117  020105
 4447  021712  054450  047440  020122
 4448  021720  024516  037440  000040
 4449  021726  050117  051105  052101  MSEC22: .ASCIZ  /OPERATION WILL BE IN 32 SECTOR (16 BIT) MODE/<CR><LF>
 4450  021734  047511  020116  044527
 4451  021742  046114  041040  020105
 4452  021750  047111  031440  020062
 4453  021756  042523  052103  051117
 4454  021764  024040  033061  041040
 4455  021772  052111  020051  047515
 4456  022000  042504  005015     000
 4457  022005     117  042520  040522  MSEC20: .ASCIZ  /OPERATION WILL BE IN 30 SECTOR (18 BIT) MODE/<CR><LF/
 4458  022012  044524  047117  053440
 4459  022020  046111  020114  042502
 4460  022026  044440  020116  030063
 4461  022034  051440  041505  047524
 4462  022042  020122  030450  020070
 4463  022050  044502  024524  046440
 4464  022056  042117  006505  000012
 4465  022064  047111  040526  044514  BADENT: .ASCIZ  /INVALID ENTRY/<CR><LF>
 4466  022072  020104  047105  051124
 4467  022100  006531  000012
 4468  022104  047105  044504  043516  MADRER: .ASCII  /ENDING DSK ADRS MUST BE EQUAL TO OR GREATER/<CR><LF>
 4469  022112  042040  045523  040440
 4470  022120  051104  020123  052515
 4471  022126  052123  041040  020105
 4472  022134  050505  040525  020114
 4473  022142  047524  047440  020122
 4474  022150  051107  040505  042524
 4475  022156  006522     012
 4476  022161     124  040510  020116          .ASCIZ  /THAN STARTING ADRS/<CR><LF>
 4477  022166  052123  051101  044524
 4478  022174  043516  040440  051104
```

```
 4479  022202  006523  000012
 4480  022206  042523  042514  052103  MSELP:  .ASCII  /SELECT DATA PATTERN (BY ENTERING 0,1 OR 2)/<CR><LF>
 4481  022214  042040  052101  020101
 4482  022222  040520  052124  051105
 4483  022230  020116  041050  020131
 4484  022236  047105  042524  044522
 4485  022244  043516  030040  030454
 4486  022252  047440  020122  024462
 4487  022260  005015
 4488  022262  024040  024460  055040          .ASCII  / (0) ZERO'S AND ONE'S/<CR><LF>
 4489  022270  051105  023517  020123
 4490  022276  047101  020104  047117
 4491  022304  023505  006523     012
 4492  022311     040  030450  020051          .ASCII  / (1) AB AND CD/<CR><LF>
 4493  022316  041101  040440  042116
 4494  022324  041440  006504     012
 4495  022331     040  031050  020051          .ASCIZ  / (2) WORST CASE: /
 4496  022336  047527  051522  020124
 4497  022344  040503  042523  020072
 4498  022352     000
 4499
 4500  022353     127  051117  052123  MPATD:  .ASCIZ  /WORST CASE/
 4501  022360  041440  051501  000105
 4502  022366  005015  051412  040524  MSFOU:  .ASCIZ  <CR><LF><LF>/STARTING FORMAT ON DRIVE /
 4503  022374  052122  047111  020107
 4504  022402  047506  046522  052101
 4505  022410  047440  020116  051104
 4506  022416  053111  020105     000
 4507  022423     015  005012  052123  MSCHK:  .ASCIZ  <CR><LF><LF>/STARTING CHECK ON DRIVE /
 4508  022430  051101  044524  043516
 4509  022436  041440  042510  045503
 4510  022444  047440  020116  051104
 4511  022452  053111  020105     000
 4512  022457     015  005012  047506  MFCMPT: .ASCIZ  <CR><LF><LF>/FORMAT COMPLETE. /
 4513  022464  046522  052101  041440
 4514  022472  046517  046120  052105
 4515  022500  026105  000040
 4516  022504  005015  041412  042510  MCCMPT: .ASCIZ  <CR><LF><LF>/CHECK COMPLETE.
 4517  022512  045503  041440  046517
 4518  022520  046120  052105  026105
 4519  022526  000040
 4520  022530  047524  040524  020114  NUMERR: .ASCIZ  /TOTAL ERRORS DETECTED: /
 4521  022536  051105  047522  051522
 4522  022544  042040  052105  041505
 4523  022552  042524  035104  000040
 4524  022560  051120  051505  047105  ADDRIS: .ASCIZ  /PRESENT ADDRESS IS: /
 4525  022566  020124  042101  051104
 4526  022574  051505  020123  051511
 4527  022602  020072     000
 4528
 4529                                  ;;*****************************************************************
 4530
                                       .SBTTL  ERROR MESSAGES
                                       ;;*****************************************************************
```

```
4535  022605     122  030510  020061  EM1:    .ASCIZ  /RH11 INTERRUPT OCCURRED (RMAS=0)/
4536  022612  047111  042524  051122
4537  022620  050125  020124  041517
4538  022626  052503  051122  042105
4539  022634  024040  046522  051501
4540  022642  030075  000051
4541  022646  047125  054105  042520  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
4542  022654  052103  042105  040440
4543  022662  052124  047105  044524
4544  022670  047117  047440  041503
4545  022676  051125  042522  000104
4546  022704  040515  051523  052502  EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
4547  022712  020123  040520  044522
4548  022720  054524  042440  051122
4549  022726  051117  024040  041515
4550  022734  042520  030475  000051
4551  022742  040515  051523  052502  EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
4552  022750  020123  040520  044522
4553  022756  054524  042440  051122
4554  022764  051117  024040  040520
4555  022772  036522  024461     000
4556  022777     101  042104  042522  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
4557  023004  051523  050040  052514
4558  023012  020107  044103  047101
4559  023020  042507  041040  052111
4560  023026  051440  052105     000
4561  023033     122  030510  020061  EM6:    .ASCIZ  /RH11 DIDN'T RESPOND TO ADDRESSING
4562  023040  044504  047104  052047
4563  023046  051040  051505  047520
4564  023054  042116  052040  020117
4565  023062  042101  051104  051505
4566  023070  044523  043516     000
4567  023075     104  044522  042526  EM10:   .ASCIZ  /DRIVE OFFLINE/
4568  023102  047440  043106  044514
4569  023110  042516     000
4570  023113     120  051105  044523  EM11:   .ASCIZ  /PERSISTENT DRIVE UNSAFE ERROR/
4571  023120  052123  047105  020124
4572  023126  051104  053111  020105
4573  023134  047125  040523  042506
4574  023142  042440  051122  051117
4575  023150     000
4576  023151     125  041516  051117  EM12:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
4577  023156  042522  052103  041101
4578  023164  042514  046440  051501
4579  023172  041123  051525  050040
4580  023200  051101  052111  020131
4581  023206  051105  047522  000122
4582  023214  047523  052106  040527  EM13:   .ASCIZ  /SOFTWARE TIMEOUT/
4583  023222  042522  052040  046511
4584  023230  047505  052125     000
4585  023235     104  044522  042526  EM14:   .ASCIZ  /DRIVE UNSAFE ERROR/
4586  023242  052440  051516  043101
4587  023250  020105  051105  047522
4588  023256  000122
4589  023260  047503  052116  047522  EM15:   .ASCIZ  &CONTROLLER/DRIVE ERROR DURING WRITE&
4590  023266  046114  051105  042057
```

```
 4591  023274  044522  042526  042440
 4592  023302  051122  051117  042040
 4593  023310  051125  047111  020107
 4594  023316  051127  052111  000105
 4595  023324  047503  052116  047522  EM16:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE CHECK@
 4596  023332  046114  051105  042057
 4597  023340  044522  042526  042440
 4598  023346  051122  051117  042040
 4599  023354  051125  047111  020107
 4600  023362  051127  052111  020105
 4601  023370  044103  041505  000113
 4602  023376  042522  051124  042511  EM17:   .ASCIZ  @RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE@
 4603  023404  020123  047516  020124
 4604  023412  052523  042503  051523
 4605  023420  052506  020114  020055
 4606  023426  042523  052103  051117
 4607  023434  047040  052117  040440
 4608  023442  041503  050105  040524
 4609  023450  046102  000105
 4610  023454  040504  040524  042440  EM20:   .ASCIZ  @DATA ERROR DURING WRITE CHECK@
 4611  023462  051122  051117  042040
 4612  023470  051125  047111  020107
 4613  023476  051127  052111  020105
 4614  023504  044103  041505  000113
 4615  023512  047503  052116  047522  EM21:   .ASCIZ  @CONTROLLER/DRIVE ERROR VERIFYING HEADERS@
 4616  023520  046114  051105  042057
 4617  023526  044522  042526  042440
 4618  023534  051122  051117  053040
 4619  023542  051105  043111  044531
 4620  023550  043516  044040  040505
 4621  023556  042504  051522     000
 4622  023563     110  040505  042504  EM22:   .ASCIZ  @HEADER COMPARE ERROR VERIFYING HEADERS@
 4623  023570  020122  047503  050115
 4624  023576  051101  020105  051105
 4625  023604  047522  020122  042526
 4626  023612  044522  054506  047111
 4627  023620  020107  042510  042101
 4628  023626  051105  000123
 4629  023632  054503  044514  042116  EM23:   .ASCIZ  @CYLINDER FIELD IN HEADER IS NOT CORRECT@
 4630  023640  051105  043040  042511
 4631  023646  042114  044440  020116
 4632  023654  042510  042101  051105
 4633  023662  044440  020123  047516
 4634  023670  020124  047503  051122
 4635  023676  041505  000124
 4636  023702  051122  052111  020105  EM24:   .ASCIZ  @WRITE CHECK ERROR@
 4637  023710  044103  041505  020113
 4638  023716  051105  047522  000122
 4639
 4640                          ;;*****************************************************************
 4641
 4642  023724  046522  051501  051011  DH1:    .ASCIZ  /RMAS   RMCS1   RMER1   RMER2   RMDS    RMDC    RMDA/
 4643  023732  041515  030523  051011
 4644  023740  042515  030522  051011
 4645  023746  042515  031122  051011
 4646  023754  042115  004523  046522
```

```
 4647  023762  041504  051011  042115
 4648  023770  000101
 4649  023772  051104  053111  020105  DH2:    .ASCIZ  /DRIVE   RMDS   RMER1   RMER2   RMAS    RMCS1/
 4650  024000  020040  046522  051504
 4651  024006  020040  051040  042515
 4652  024014  030522  020040  051040
 4653  024022  042515  031122  020040
 4654  024030  051040  040515  004523
 4655  024036  046522  051503  000061
 4656  024044  051104  053111  020105  DH3:    .ASCIZ  /DRIVE   REG ADR  DATA  RMCS1   RMER1   RMER2/
 4657  024052  020040  042522  020107
 4658  024060  042101  020122  042040
 4659  024066  052101  020101  051040
 4660  024074  041515  030523  051011
 4661  024102  042515  030522  051011
 4662  024110  042515  031122     000
 4663  024115     104  044522  042526  DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD    BAD   RMCS1 RMER1   RMER2/
 4664  024122  020040  051040  043505
 4665  024130  040440  051104  020040
 4666  024136  020040  047507  042117
 4667  024144  020040  020040  040502
 4668  024152  020104  020040  046522
 4669  024160  051503  004461  046522
 4670  024166  051105  004461  046522
 4671  024174  051105  000062
 4672  024200  044122  040440  042104  DH6:    .ASCIZ  /RH ADDRESS/
 4673  024206  042522  051523     000
 4674  024213     104  044522  042526  DH10:   .ASCIZ  /DRIVE   ERR PC  RMCS1   RMCS2   RMDS    RMER1   RMMR2/
 4675  024220  020040  042440  051122
 4676  024226  050040  020103  051040
 4677  024234  041515  030523  020040
 4678  024242  051040  041515  031123
 4679  024250  020040  051040  042115
 4680  024256  020123  020040  046522
 4681  024264  051105  020061  020040
 4682  024272  046522  051115  000062
 4683  024300  046522  041505  020061  DH10A:  .ASCIZ  /RMEC1   RMEC2   RMWC    RMBA    RMDA    RMAS    RMLA/
 4684  024306  020040  046522  041505
 4685  024314  020062  020040  046522
 4686  024322  041527  020040  020040
 4687  024330  046522  040502  020040
 4688  024336  020040  046522  040504
 4689  024344  020040  020040  046522
 4690  024352  051501  020040  020040
 4691  024360  046522  040514     000
 4692  024365     122  042115  004502  DH10B:  .ASCIZ  /RMDB    RMMR    RMDT    RMSN    RMOF  CYLINDER   TRACK/
 4693  024372  046522  051115  051011
 4694  024400  042115  004524  046522
 4695  024406  047123  004440  046522
 4696  024414  043117  020040  054503
 4697  024422  044514  042116  051105
 4698  024430  020040  052040  040522
 4699  024436  045503     000
 4700  024441     104  044522  042526  DH17:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
 4701  024446  020040  042440  051122
 4702  024454  050040  020103  054503
```

```
 4703  024462  044514  042116  051105
 4704  024470  052040  040522  045503
 4705  024476  020040  051440  041505
 4706  024504  047524  000122
 4707  024510  051104  053111  020105  DH20:   .ASCIZ  /DPIVE   ERR PC CYLINDER TRACK   SECTOR/
 4708  024516  020040  051105  020122
 4709  024524  041520  041440  046131
 4710  024532  047111  042504  020122
 4711  024540  051124  041501  020113
 4712  024546  020040  042523  052103
 4713  024554  051117     000
 4714  024557     122  041515  030523  DH20A:  .ASCIZ  /RMCS1   RMCS2   RMDS   RMER1   RMER2   RMMR2   RMEC1   RMEC2/
 4715  024564  020040  051040  041515
 4716  024572  031123  020040  051040
 4717  024600  042115  020123  020040
 4718  024606  046522  051105  020061
 4719  024614  020040  046522  051105
 4720  024622  020062  020040  046522
 4721  024630  051115  020062  020040
 4722  024636  046522  041505  020061
 4723  024644  020040  046522  041505
 4724  024652  000062
 4725  024654  046522  041527  020040  DH20B:  .ASCIZ  /RMWC    RMBA    RMDA    RMAS    RMLA    RMDB    RMMR1   RMDT/
 4726  024662  020040  046522  040502
 4727  024670  020040  020040  046522
 4728  024676  040504  020040  020040
 4729  024704  046522  051501  020040
 4730  024712  020040  046522  040514
 4731  024720  020040  020040  046522
 4732  024726  041104  020040  020040
 4733  024734  046522  051115  020061
 4734  024742  020040  051040  042115
 4735  024750  000124
 4736  024752  046522  047123  051011  DH20C:  .ASCIZ  /RMSN   RMOF  CYLINDER  TRACK/
 4737  024760  047515  020106  041440
 4738  024766  046131  047111  042504
 4739  024774  020122  052040  040522
 4740  025002  045503     000
 4741  025005     040  020040  020040  DH23:   .ASCII  /                EXPT'D/<CR><LF>
 4742  025012  020040  020040  020040
 4743  025020  020040  020040  042440
 4744  025026  050130  023524  006504
 4745  025034     012
 4746  025035     104  044522  042526          .ASCIZ  /DRIVE   ERR PC  CYLNDR  ACTLAL HEADER/
 4747  025042  020040  042440  051122
 4748  025050  050040  020103  041440
 4749  025056  046131  042116  020122
 4750  025064  040440  052103  040525
 4751  025072  020114  042510  042101
 4752  025100  051105     000
 4753  025103     040  020040  020040  DH24:   .ASCII  /                                        MEMORY  DISK/<CR><LF>
 4754  025110  020040  020040  020040
 4755  025116  020040  020040  020040
 4756  025124  020040  020040  020040
 4757  025132  020040  020040  020040
 4758  025140  020040  020040  020040
```

```
 4759  025146  020040  020040  046440
 4760  025154  046505  051117  020131
 4761  025162  042040  051511  006513
 4762  025170     012
 4763  025171     104  044522  042526          .ASCIZ  /DRIVE   ERR PC  CYLNDR  TRACK   SECTOR  DATA    DATA/
 4764  025176  020040  042440  051122
 4765  025204  050040  020103  041440
 4766  025212  046131  042116  020122
 4767  025220  052040  040522  045503
 4768  025226  020040  051440  041505
 4769  025234  047524  020122  042040
 4770  025242  052101  020101  020040
 4771  025250  042040  052101  000101
 4772
 4773
 4774  025256  001276  001306  001322  DT1:    .WORD   ATTN,RM.REG,RM.REG+14,RM.REG+40,RM.REG+6,RM.REG+34,RM.REG+6
 4775  025264  001346  001314  001342
 4776  025272  001314
 4777  025274  001274  012546  012550  DT2:    .WORD   DDRIVE,RMERRS,RMERRS+2,RMERRS+4,ATTN,RMCS1
 4778  025302  012552  001276  000000
 4779  025310  001274  017676  017700  DT3:    .WORD   DDRIVE,RD.ADR,RD.WRD,RM.REG,RM.REG+14,RM.REG+40
 4780  025316  001306  001322  001346
 4781  025324  001274  020122  020120  DT4:    .WORD   DDRIVE,WRT.AD,WRT.WD,RD.WRD,RM.REG,RM.REG+14,RM.REG+40
 4782  025332  017700  001306  001322
 4783  025340  001346
 4784  025342  001172                  DT6:    .WORD   $RMADR
 4785  025344  001214  001116  001306  DT10:   .WORD   DRIVE,$ERRPC,RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42
 4786  025352  001316  001320  001322
 4787  025360  001350
 4788  025362  001352  001354  001310          .WORD   RM.REG+44,RM.REG+46,RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20
 4789  025370  001312  001314  001324
 4790  025376  001326
 4791  025400  001330  001332  001334          .WORD   RM.REG+22,RM.REG+24,RM.REG+26,RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
 4792  025406  001336  001340  001270
 4793  025414  001272
 4794  025416  001214  001116  001270  DT17:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
 4795  025424  001272  001252
 4796  025430  001214  001116  001270  DT20:   .WORD   DRIVE,$ERRPC,DS.CYL,DS TRK,SAVSEC
 4797  025436  001272  001252
 4798  025442  001306  001316  001320          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+40,RM.REG+42,RM.REG+44,RM.RE
 4799  025450  001322  001346  001350
 4800  025456  001352  001354
 4801  025462  001310  001312  001314          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.RE
 4802  025470  001324  001326  001330
 4803  025476  001332  001334
 4804  025502  001336  001340  001270          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
 4805  025510  001272
 4806  025512  001214  001116  025732  DT23:   .WORD   DRIVE,$ERRPC,BUFP,RBUF,RBUF+2
 4807  025520  025722  025724
 4808  025524  001214  001116  001270  DT24:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC,$GDDAT,RM.REG+22
 4809  025532  001272  001252  001124
 4810  025540  001330
 4811  025542  001306  001316  001320          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+40,RM.REG+42,RM.REG+44,RM.RE
 4812  025550  001322  001346  001350
 4813  025556  001352  001354
 4814  025562  001310  001312  001314          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.RE
```

```
 4815  025570  001324  001326  001330
 4816  025576  001332  001334
 4817  025602  001336  001340  001270          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
 4818  025610  001272
 4819
 4820  025612  000001                  DF1:    .WORD   1
 4821  025614     007     000                  .BYTE   7,0
 4822  025616  000001                  DF2:    .WORD   1
 4823  025620     006     000                  .BYTE   6,0
 4824  025622  000001                  DF3:    .WORD   1
 4825  025624     006     000                  .BYTE   6,0
 4826  025626  000001                  DF4!    .WORD   1
 4827  025630     007     000                  .BYTE   7,0
 4828  025632  000001                  DF6:    .WORD   1
 4829  025634     001     000                  .BYTE   1,0
 4830  025636  000003                  DF10:   .WORD   3
 4831  025640     007     000                  .BYTE   7,0
 4832  025642  024300                          .WORD   DH10A
 4833  025644     007     000                  .BYTE   7,0
 4834  025646  024365                          .WORD   DH10B
 4835  025650     007     140                  .BYTE   7,140
 4836  025652  000001                  DF17:   .WORD   1
 4837  025654     005     034                  .BYTE   5,34
 4838  025656  000004                  DF20:   .WORD   4
 4839  025660     005     034                  .BYTE   5,34
 4840  025662  024557                          .WORD   DH20A
 4841  025664     010     000                  .BYTE   8.,0
 4842  025666  024654                          .WORD   DH20B
 4843  025670     010     000                  .BYTE   8.,0
 4844  025672  024752                          .WORD   DH20C
 4845  025674     004     014                  .BYTE   4,14
 4846  025676  000001                  DF23:   .WORD   1
 4847  025700     005     000                  .BYTE   5,0
 4848  025702  000004                  DF24:   .WORD   4
 4849  025704     007     034                  .BYTE   7,34
 4850  025706  024557                          .WORD   DH20A
 4851  025710     010     000                  .BYTE   8.,0
 4852  025712  024654                          .WORD   DH20B
 4853  025714     010     000                  .BYTE   8.,0
 4854  025716  024752                          .WORD   DH20C
 4855  025720     004     014                  .BYTE   4,14
 4856
 4857                                  ;;*****************************************************************
 4858
 4859                                  ;BUFFER STARTS HERE
 4860
 4861                                  ;;*****************************************************************
 4862
 4863  025722  000000  000000  000000  RBUF:   .WORD   0,0,0,0          ;BUFFER FOR HEADER CHECK
 4864  025730  000000
 4865
 4866  025732                          BUFP:                   ;FORMAT AND CHECK BUFFER STARTS HERE
 4867                                                          ;AND WILL OVERLAY ALL REMAINING CODE IN PROGRAM
 4868
 4869  025732  005015  046412  044501  TITLE:  .ASCII  <CR><LF><LF>/MAINDEC-11-TEMP<CR><LF>
 4870  025740  042116  041505  030455
```

```
 4871  025746  026461  042524  050115
 4872  025754  005015
 4873  025756  046522  030460  043040          .ASCIZ  @RM01 FORMATTER PROGRAM@<CR><LF><LF>
 4874  025764  051117  040515  052124
 4875  025772  051105  050040  047522
 4876  026000  051107  046501  005015
 4877  026006  000012
 4878  026010  005015  047524  023440  LOADRV: .ASCII  <CR><LF>/TO 'FORMAT' OR 'CHECK' DRIVE 0, REPLACE THE 'XXDP'/<CR><LF>
 4879  026016  047506  046522  052101
 4880  026024  020047  051117  023440
 4881  026032  044103  041505  023513
 4882  026040  042040  044522  042526
 4883  026046  030040  020054  042522
 4884  026054  046120  041501  020105
 4885  026062  044124  020105  054047
 4886  026070  042130  023520  005015
 4887  026076  040520  045503  047440          .ASCII  /PACK ON DRIVE 0 WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40./<CR><LF>
 4888  026104  020116  051104  053111
 4889  026112  020105  020060  044527
 4890  026120  044124  040440  047516
 4891  026126  044124  051105  050040
 4892  026134  041501  026113  041440
 4893  026142  042514  051101  046440
 4894  026150  046505  051117  020131
 4895  026156  047514  040503  044524
 4896  026164  047117  032040  026060
 4897  026172  005015
 4898  026174  047101  020104  042522          .ASCIZ  /AND RESTART THE PROGRAM/<CR><LF>
 4899  026202  052123  051101  020124
 4900  026210  044124  020105  051120
 4901  026216  043517  040522  006515
 4902  026224  000012
 4903
 4904
 4905                                  .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
 4906                                  ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
 4907                                  ;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
 4908                                  ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 4909                                  ;REQUIRED.
 4910                                  ;NOTE: THIS ROUTINE DESTROYS R0-R4
 4911                                  ;CALL
 4912                                  ;
 4913                                  ;       JSR     PC,BUSADR
 4914                                  ;       RETURN
 4915
 4916  026226  005737  001302          BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
 4917  026232  001450                          BEQ     7$              ;NO--BRANCH
 4918  026234  005037  001302                  CLR     CHGADR          ;YES--CLEAR THE REQUEST FLAG
 4919  026240  012700  001172          1$:     MOV     #$RMADR,R0      ;FIRST ADDRESS
 4920  026244  104401  026426                  TYPE    ,MRMCS1         ;"RMCS1="
 4921  026250  012046                          MOV     (R0)+,-(SP)     ;PRESENT RMCS1 ADDRESS
 4922  026252  104402                          TYPOC                   ;TYPE IT
 4923  026254  104401  021337                  TYPE    ,LINSP          ;2 SPACES
 4924  026260  104411                          RDLIN                   ;GET THE ENTRY
 4925  026262  012601                          MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
 4926  026264  004537  026450                  JSR     R5,CK.NUM       ;CHECK THE NUMBER
```

```
4927  026270  026310                        3$                      ;CARRIAGE RETURN ONLY ENTERED
4928  026272  026354                        7$                      ;PERIOD ONLY ENTERED
4929  026274  026240                        1$                      ;ILLEGAL INPUT
4930  026276  026304                        2$                      ;TERMINATED WITH A CARRIAGE RETURN
4931  026300  026240                        1$                      ;TERMINATED WITH A ","
4932  026302  026350                        4$                      ;TERMINATED WITH A "."
4933  026304  010260  177776        2$:     MOV     R2,-2(R0)       ;SAVE NEW RMCS1
4934  026310  104401  026437        3$:     TYPE    ,MRHVEC         ;"RHVEC="
4935  026314  012046                        MOV     (R0)+,-(SP)     ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
4936  026316  104402                        TYPOC                   ;TYPE IT
4937  026320  104401  021337                TYPE    ,LINSP          ;2 SPACES
4938  026324  104411                        RDLIN                   ;READ THE ENTRY
4939  026326  012601                        MOV     (SP)+,R1        ;ASCII TEXT ADDRESS
4940  026330  004537  026450                JSR     R5,CK.NUM       ;CHECK THE NUMBER
4941  026334  026354                        7$                      ;CARRIAGE RETURN ONLY ENTERED
4942  026336  026354                        7$                      ;PERIOD ONLY ENTERED
4943  026340  026310                        3$                      ;ILLEGAL INPUT
4944  026342  026350                        4$                      ;TERMINATED WITH A CARRIAGE RETURN
4945  026344  026310                        3$                      ;TERMINATED WITH A ","
4946  026346  026350                        4$                      ;TERMINATED WITH A "."
4947  026350  010260  177776        4$:     MOV     R2,-2(R0)               ;SAVE INPUT
4948  026354  013701  000004        7$:     MOV     ERRVEC,R1       ;SAVE THE ERROR VECTOR
4949  026360  012737  026414  000004        MOV     #8$,ERRVEC      ;SETUP FOR TRAP
4950  026366  005777  152600                TST     @$RMADR         ;CHECK FOR RH11
4951  026372  010137  000004                MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
4952  026376  012700  001172                MOV     #$RMADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
4953  026402  012701  012714                MOV     #RMADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
4954  026406  012021                        MOV     (R0)+,(R1)+     ;BUS ADDRESS
4955  026410  012021                        MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
4956  026412  000207                        RTS     PC              ;RETURN
4957  026414  010137  000004        8$:     MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
4958  026420  022626                        CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
4959  026422  104006                        ERROR   6               ;REPORT THE ERROR
4960  026424  000705                        BR      1$              ;ASK FOR BUS ADDRESS
4961
4962  026426  046522  051503  020061  MRMCS1: .ASCIZ  @RMCS1 = @
4963  026434  020075     000
4964  026437     122  053110  041505  MRHVEC: .ASCIZ  @RHVEC = @
4965  026444  036440  000040
4966
4967                                  .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
4968                                  ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
4969                                  ;AND FORMS AN OCTAL NUMBER IN R2
4970                                  ;CALL:
4971                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
4972                                  ;       MOV     #NUM,R2         ;MAX SIZE OF INPUT NUMBER
4973                                  ;       JSR     R5,CK.NUM       ;GO FORM THE NUMBER
4974                                  ;       RETURN  ADR1            ;"CR" ONLY ENTERED -- R2 = 0
4975                                  ;       RETURN  ADR2            ;"PERIOD" ONLY ENTERED -- R2 = 0
4976                                  ;       RETURN  ADR3            ;ILLEGAL CHARACTER IN THE INPUT STRING
4977                                  ;       RETURN  ADR4            ;"CR" ENTERED -- R2 = NUMBER
4978                                  ;       RETURN  ADR5            ;"COMMA" -- R2 = NUMBER
4979                                  ;       RETURN  ADR6            ;"PERIOD" -- R2 = NUMBER
4980
4981  026450  010446                  CK.NUM: MOV     R4,-(SP)        ;SAVE R4
4982  026452  010346                          MOV     R3,-(SP)        ;SAVE R3
```

```
 4983  026454  010246                        MOV     R2,-(SP)        ;SAVE R2
 4984  026456  005004                        CLR     R4              ;RETURN POINTER
 4985  026460  005003                        CLR     R3              ;START NUMBER AT ZERO
 4986  026462  005002                        CLR     R2              ;STORE RESULT
 4987  026464  004537  007074                JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 4988  026470  026566                        6$                      ;ILLEGAL CHARACTER
 4989  026472  026572                        8$                      ;CARRIAGE RETURN
 4990  026474  026566                        6$                      ;" "
 4991  026476  026570                        7$                      ;"."
 4992  026500  026504                        1$                      ;DIGIT 0-7
 4993  026502  026566                        6$                      ;DIGIT 8-9
 4994  026504  062705  000004        1$:     ADD     #4,R5           ;INCREMENT RETURN PAST "CR" AND "PERIOD" ONLY RETURNS
 4995  026510  006303        2$:     ASL     R3                      ;FOR THE OCTAL NUMBER IN R3
 4996  026512  103425                BCS     6$                      ;DON'T LET IT GET TO BIG
 4997  026514  006303                ASL     R3
 4998  026516  103423                BCS     6$
 4999  026520  006303                ASL     R3
 5000  026522  103421                BCS     6$
 5001  026524  060203                ADD     R2,R3
 5002  026526  004537  007074        3$:     JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 5003  026532  026572                        8$                      ;ILLEGAL CHARACTER
 5004  026534  026556                        5$                      ;CARRIAGE RETURN
 5005  026536  026554                        4$                      ;" "
 5006  026540  026546                        3$                      ;"."
 5007  026542  026510                        2$                      ;DIGIT 0-7
 5008  026544  026572                        8$                      ;DIGIT 8-9
 5009  026546  105711                TSTB    (R1)                    ;DOES A "CR" FOLLOW THE "PERIOD"
 5010  026550  001010                BNE     8$                      ;BR IF NOT
 5011  026552  005724                TST     (R4)+                   ;INCREMENT THE RETURN
 5012  026554  005724        4$:     TST     (R4)+                   ;INCREMENT THE RETURN INDEX
 5013  026556  005724        5$:     TST     (R4)+                   ;INCREMENT THE RETURN INDEX
 5014  026560  020316                CMP     R3,(SP)                 ;INPUT VALUE TOO LARGE ?
 5015  026562  101003                BHI     8$                      ;BR IF IT IS
 5016  026564  000401                BR      7$                      ;BR IF NOT
 5017  026566  005725        6$:     TST     (R5)+                   ;INCREMENT THE RETURN ADDRESS
 5018  026570  005725        7$:     TST     (R5)+                   ;INCREMENT THE RETURN ADDRESS
 5019  026572  060405        8$:     ADD     R4,R5                   ;SETUP FOR PROPER RETURN
 5020  026574  010302                MOV     R3,R2                   ;LOAD ENTERED VALUE
 5021  026576  005726                TST     (SP)+                   ;CLEAN OFF THE STACK
 5022  026600  012603                MOV     (SP)+,R3                ;RESTORE R3
 5023  026602  012604                MOV     (SP)+,R4                ;RESTORE R4
 5024  026604  011505                MOV     (R5),R5                 ;GET RETURN ADDRESS
 5025  026606  000205                RTS     R5                      ;RETURN
 5026
 5027
 5028
 5029  000000:                .END
```

```
ACK    = 000123            829#
ACTDRV  012632            3070#   3338*   3389*   3708*   3717*   3946
ACTSTR  012633            3076#   3948*   3963*
ADDRIS  022560            1991    4524#
ADRTBL  001500             993#   1644
AOE    = 001000             731#
ATA    = 100000             718#
ATABIT  012702            3142#   3305    3430    3466    3508    3871    3890    3898    3899    3916    4022
ATTN    001276             922#   2189*   4774    4777
AT0    = 000001             751#
AT1    = 000002             752#
AT2    = 000004             753#
AT3    = 000010             754#
AT4    = 000020             755#
AT5    = 000040             756#
AT6    = 000100             757#
AT7    = 000200             758#
A16    = 000400             655#
A17    = 001000             656#
BADENT  022064            2033    4465#
BAI    = 000010             673#
BEGCYL  001224             898#    974    1385*   1441*   1449    1500    1509    1513    1540    1685    1689    1726    1729*
                          1732*   1836    1858    1860*   1867*   1888
BEGIN   002140              532    1224#
BEGIN1  002150              529    1226#   1261
BEGIN2  002154             1225    1227#
BEGTRK  001230              900#    975    1384*   1440*   1453    1503    1516    1727    1728*   1731*   1835    1838    1859
                           1861*   1866*   1889
BIT0   = 000001              629#
BIT00  = 000001              619#    629    1323    1437    1540    1914
BIT01  = 000002              618#    628    1326    1437    1821    3381    3630
BIT02  = 000004              617#    627
BIT03  = 000010              616#    626    3830    4144
BIT04  = 000020              615#    625    1802    3865
BIT05  = 000040              614#    624    3232    3650    3845    3996
BIT06  = 000100              613#    623    3314    3372    3447    3743    4205
BIT07  = 000200              612#    622    3314    3582    3729    3830    3845    3865    3894    4177
BIT08  = 000400              611#    621    1799    3314    4009
BIT09  = 001000              610#    620    1799    2216    3994
BIT1   = 000002              628#   1671    1790
BIT10  = 002000              609#   1796    1821    2200    3632
BIT11  = 004000              608#   1796    3279    3422    3601    3917
BIT12  = 010000              607#   1793    1821    3274    3296    3314    3383    3404    3416    3597    3616    3629    3641
                            3855    3857    4082    4139    4206
BIT13  = 020000              606#   1790    1821    2207    3366    3803    4085
BIT14  = 040000              605#   1790    1821    3378    3413    3790    3933    4049    4094    4099    4203
BIT15  = 100000              604#   3366    3378    3381    3383    3413    3416    3601    3630    3632    3743    3930    3845
                            3855    3865    3933    3994    4009    4049    4056
BIT2   = 000004              627#   1790    4056
BIT3   = 000010              626#
BIT4   = 000020              625#
BIT5   = 000040              624#
BIT6   = 000100              623#   3408
BIT7   = 000200              622#
BIT8   = 000400              621#
BIT9   = 001000              620#
```

RMFOR.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
BPTVEC= 000014            636#
BUFP    025732            993    994    995    996    997    998    999   1000   1001   1002   1003   1004   1005
                         1006   1007   1008   1009   1010   1011   1012   1013   1014   1015   1016   1017   1018
                         1019   1020   1021   1022   1023   1024   1537   1572   1712   1933   1887   1905   1932
                         4806   4866#
BUSADR  026226           1281   4916#
C       021327           1992   4400#
CHGADR  001302            925#  1224#  1226#  4916   4918#
CI1     014212           3437   3465#
CI3     014326           3439   3489#
CI4     014434           3426   3510#
CI5     015012           3488   3509   3587#  3596
CI6     015034           3526   3536   3538   3540   3592#
CI7     015050           3368   3446   3478   3482   3486   3494   3503   3507   3518   3525   3531   3535   3549
                         3555   3559   3569   3580   3595   3597#  3703   3733   3882   4189
CI7B    015076           3600   3603#  3884
CI8     015150           3598   3606   3615#  3756   4041
CKADRS  003652           1500#
CKSWR = 104407           2187   2215   2958#
CKTRK   004406           1574   1602#
CK.CHR  007074           1400   1465   2093#  2130   2144   4987   5002
CK.DEC  007046           2071#  2099
CK.DIG  007146           2024   2124#
CK.NUM  026450           4926   4940   4981#
CK.OCT  007020           2053#  2101
CLOCK   006514           1946   1956   1967#
CLR   = 000040            675#
CLRQUE  020706           3214   3649   4019   4273#
CNTLC   001300            924#  1261#  1394#  1990#  2553   2636
CR    = 000015            544#   865#  2354   2364   4394   4397   4405   4409   4412   4416   4420   4424   4429
                         4442   4449   4457   4465   4468   4476   4480   4488   4492   4502   4507   4513   4516
                         4741   4753   4869   4873   4878   4887   4898
CRLF  = 000200            545#  2325   2364
CYLCK   001246            907#  1685#  1716   1719#  1720   1722#
DCK   = 100000            737#  1626
DDISP = 177570            551#   870   1247
DDRIVE  001274            921#  2188#  4777   4779   4781
DF1     025612           1083   4820#
DF10    025636           1132   1139   1146   1153   1160   1167   1174   1195   1202   4830#
DF17    025652           1181   4836#
DF2     025616           1090   4822#
DF20    025656           1188   4838#
DF23    025676           1209   4846#
DF24    025702           1216   4848#
DF3     025622           1097   4824#
DF4     025626           1104   4826#
DF6     025632           1118   4828#
DH1     023724           1081   4642#
DH10    024213           1130   1137   1144   1151   1158   1165   1172   1193   1200   4674#
DH10A   024300           4683#  4832
DH10B   024365           4692#  4834
DH17    024441           1179   4700#
DH2     023772           1088   4649#
DH20    024510           1186   4707#
DH20A   024557           4714#  4840   4850
DH20B   024654           4725#  4842   4852
```

```
DH2OC   024752          4736*   4844    4854
DH23    025005          1207    4741*
DH24    025103          1214    4753*
DH3     024044          1095    4656*
DH4     024115          1102    4663*
DH6     024200          1116    4672*
DISPLA  001142           870*   1247*   1255*   2199*
DISPRE  000174           513*   1255
CLT   = 100000           685*
DMD   = 000001           741*
DONE    005446          1772    1773*
DPINT   012606          3030*   3243    3246    3322*   3348    3774    3799    3923    3925*   3986    4024    4035    4043*
DPR   = 000400           711*
DPRQS   012616          3043*   3354    3441*   3777    3937*   3988    4026    4037    4052*
DRIVE   001214           894*   1412*   1433    1435    1491    1830    4785    4794    4796    4806    4808
DRQ   = 004000           774*
DRVACT  012556          2989*   3362    3543*   3590*   3609*   3620    3635*   3722*   3795    3828    3859*   3866*   3879
                        3899*   3997*   4003    4010*
DRVCLR= 000111           824*
DRVINT  013144          3234    3268*   3350    3408    3907    3926    4001
DRVQUE  021004          3357    3370    4313*
DRVSTA  012566          1306    1422    3003*   3224*   3225*   3226*   3227*   3237*   3269*   3278*   3311*   3316*   3344
                        3352    3376    3410    3414    3544*   3653*   3780    3788    3801    3853*   3860*   3872    3929
                        4044*
DRVTYP  012576          1309    1323    1326    1427    1437    3017*   3270*   3285*   3290*   3379    3654*
DRV   = 000200           710*
DSWR  = 177570           550*    869    1246
DS.CYL  001270           919*   2190*   4791    4794    4796    4804    4808    4817
DS.TRK  001272           920*   2191*   4791    4794    4796    4804    4808    4817
DTE   = 010000           734*   1626
DTSY  = 000200           747*
DTLW    012700          3136*   3220    3428    3487*   3610    3613*   3623    3636    3638*   3647*   371:    3723*   399:
                        4006    4017*   4028    4092
DT00  = 000001           755*
DT01  = 000002           766*
DT02  = 000004           767*
DT03  = 000010           768*
DT04  = 000020           769*
DT05  = 000040           770*
DT06  = 000100           771*
DT07  = 000200           772*
DT08  = 000400           773*
DT1     025256          1082    4774*
DT10    025344          1131    1138    1145    1152    1159    1166    1173    1194    120:    4795*
DT17    025416          1180    4794*
DT2     025274          1089    4777*
DT20    025430          1187    4796*
DT23    025512          1208    4806*
DT24    025524          1215    4808*
DT3     025310          1096    4779*
DT4     025324          1103    4781*
DT5     025342          1117    4784*
EC0   = 004000           705*
EC1   = 000100           728*
EC2   = 002000           796*
EMPTYQ  020764          3607    3652    3744    3967    3935    4057    4298*
```

```
EMTVEC= 000030           639*  1236*  1237*  1260*
EM1     022605          1080   4535*
EM10    023075          1129   4567*
EM11    023113          1136   4570*
EM12    023151          1143   4576*
EM13    023214          1150   4582*
EM14    023235          1157   4585*
EM15    023260          1164   4589*
EM16    023324          1171   4595*
EM17    023376          1178   4602*
EM2     022646          1087   4541*
EM20    023454          1185   4610*
EM21    023512          1192   4615*
EM22    023563          1199   4622*
EM23    023632          1206   4629*
EM24    023702          1213   4636*
EM3     022704          1094   4546*
EM4     022742          1101   4551*
EM5     022777          4556*
EM6     023033          1115   4561*
ENDCYL  001222           897*   976   1436*  1439*  1449   1500   1510   1512   1716
ENDTRK  001226           899*   977   1387*  1453   1502   1517
ENTADR  021342          1446   4405*
ERINDX  005472          1585   1617   1700   1787*
ERR   = 040000           717*  1636
ERRVEC= 000004           632*  1244   1245*  1256*  1942*  1943*  1953*  1962*  4948   4949*  4951*  4957*
ES.SAV  021144          3759   3810   3814   4090   4150   4367*
FER   = 000020           726*  1626
FMTDPB  001356           955*  1389*  1433   1435*  1558*  1561*  1562*  1564   1566   1570*  1571*  1572*  1575*
                        1579   1561   1602*  1606   1608   1642*  1644*  1650*  1659   1686*  1687*  1688*  1689*
                        1690*  1694   1696   1706   1723*  1724*  1725*  1735*  1789   1792   1795   1798   180:
                        1820   1830*  1831*  1832*  1833*  1834*  1835*  1836*  1837*  1850*  1854*  1855*  1856
                        1863*  1864*  1865*  1876*  1877*  1878*  1993   1949   2190   219:
FM*15 = 010000           797*
F1    = 000002           700*
F2    = 000004           701*
F3    = 000010           702*
F4    = 000020           703*
F5    = 000040           704*
GETREG= 000141           831*
GETREQ  021060          3401   3625   3784   3820   3881   3932   3993   4008   4046   4053   4334*
GNS   = ****** U          512   2950   2951   2952   2953   2954   2956   2958   2959   2960   2961   2962
GO    = 000001           699*
GTSWR = 104406          1275   2956*
HCE   = 000200           729*  1626   1706
HCI   = 002000           795*
HCRC  = 000400           730*  1626
HDREAD  005020          1675   1682*
HEDERR  001262           913*  1682*
HT    = 000011           542*  2323   2364
IAE   = 002000           732*
IE    = 000100           653*
ILF   = 000001           722*
ILR   = 000002           723*
IOTVEC= 000020           637*
IS    = 000100           676*
```

```
ISR     015560          3229    3708*
LA      015404          3434    3669*
LACNT   012644          3095*   3677*   3678    3699*
LF    = 000012           543*    886*   2358    2364    4394    4397    4405    4409    4412    4416    4420    4424    4429
                        4442    4449    4457    4465    4468    4476    4480    4488    4492    4502    4507    4512    4516
                        4741    4753    4869    4873    4878    4887    4898
LINSP   021337          1321    1996    2269    2278    2289    4404*   4923    4937
LIN4SP  021335          1305    4403*
LOADRV  026010          1267    4878*
LOP.CK  005572          1592    1664    1711    1715    1816*
LST   = 002000           713*
MADRER  022104          1455    4468*
MAXSEC  001266           917*   1377*   1383*   1613    1659    1886    1904    1931
MCCMPT  022504          1750    4516*
MCLK  = 000002           742*
MCPE  = 020000           658*
MCPEMX  012712          3154*   4072    4120
MDRNP   021414          1429    4413*
MDRVD   021333          1408    4402*
MER11   021441          1431    4416*
MFCMPT  022457          1748    4512*
MFORMT  021601          1355    4434*
MHECK   021622          1353    4437*
MINX  = 000004           743*
MMODE   021547          1342    4429*
MNDLTA  012726          3170*   3697
MNRM01  021470          4420*
MODE    001220           996*   1341*   1358*   1485    1573    1629    1648    1746
MOFFLN  021372          4409*
MOH   = 020000           775*
MOL   = 010000           715*
MORMAT  021635          1357    4439*
MPATD   022353          1472    4500*
MPE   = 000400           678*
MRD   = 000020           745*
MRHVEC  026437          4934    4964*
MRMCS1  026426          4920    4962*
MSCHK   022423          1489    4507*
MSEC20  022005          1379    4457*
MSEC22  021726          1373    4449*
MSELD   021535          4427*
MSELP   022206          1462    4480*
MSFOU   022366          1487    4502*
MSIZE   021655          1362    4442*
MSTCK = 000010           744*
MUNIT   021311          1397    4397*
MUSDR   021515          1425    4424*
MWC     001260           912*   1375*   1381*   1571    1645    1832
MWR   = 000040           746*
MXDLTA  012724          3167*   3695
MXF   = 001000           679*
MXLACT  012722          3164*   3678
MXWNDW  012730          3173*   3496
MD      003174          1390    1394*   1411    1426    1430    1432    1775
MI      002716          1341*   1352
MIA     003006          1354    1362*   1372
```

```
M1B     003134          1378    1384*   1977
M2      003444          1434    1442    1446*   1456
M4      003522          1452    1454    1458    1462*   1478
M5      003604          1483*   1777
NBA   = 100000           777*
NED   = 010000           682*
NEM   = 004000           681*
NEWSVC  006546          1483    1981*   2003
NOTPRS  021226          1314    4386*
NOTRM   021211          1312    4383*
NOTSAF  021243          1318    4389*
NUMERR  022530          1751    4520*
OENTER  006526          1394    1973*   1990
OFDIR = 000001           794*
OFFSET= 000115           826*
OPI   = 020000           735*   1626
OPT     013730          3364    3399*   3739    3900    3938
OR    = 000200           677*
PAR   = 000010           725*
PARENT  006704          1447    2011*
PAR1    001430           974     982*
PAR2    001443           975     984*
PAR3    001456           976     986*
PAR4    001467           977     988*
PAT   = 000020           674*
PATA    001242           905*   1526*   1530*   1534*   1538    1542*   1544    1552*   1907    1913*   1916*   1918    1925*
PATB    001244           906*   1527*   1531*   1535*   1539    1543*   1545    1551*   1908    1912*   1917*   1919    1924*
PATSEL  001240           904*   1388*   1479*   1528    1532
PGE   = 002000           680*
PGM   = 001000           712*
PIP   = 020000           716*
PIRQ  = 177772           549*
PIRQVE= 000240           643*
PCPQUE  021102          3412    3465    3581    3893    4349*
PR0   = 000000           566*
PR1   = 000040           567*
PR2   = 000100           568*
PR3   = 000140           569*
PR4   = 000200           570*
PR5   = 000240           571*   1259    1260
PR6   = 000300           572*   1948    1958
PR7   = 000340           573*   1295
PS    = 177776           546*    547    1295*   1421*   3212    3213*   3242*   3250*   3336    3337*   3390*   3400    3450*
                        3684*
PSEL  = 002000           657*
PSW   = 177776           547*   1484*   1989*
PWRVEC= 000024           638*
QCNT    020414          4215*   4274    4298*   4313    4315*   4335    4349*
QDRV0   020506          4226    4237    4246    4258*
QDRV1   020526          4227    4238    4247    4259*
QDRV2   020546          4228    4239    4248    4260*
QDRV3   020566          4229    4240    4249    4261*
QDRV4   020606          4230    4241    4250    4262*
QDRV5   020626          4231    4242    4251    4263*
QDRV6   020646          4232    4243    4252    4264*
QDRV7   020666          4233    4244    4253    4265*
```

RMFOR.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
QINPT   020424          4226*  4300   4317*  4318*  4319   4321*
QOUTPT  020444          4237*  4300*  4338   4351   4352*  4353   4355*
QSTART  020464          4246*  4280   4285   4321   4355
QSTOP   020466          4247*  4319   4353
QTERM = 020706          4254   4266*
RBUF    025722          1687   1712   4806   4863*
RDCHR = 104410          2712   2959*
RDDAT = 000171           838*
RDHD  = 000173           839*  1690
RDLIN = 104411          1343   1363   1398   1463   2022   2960*  4924   4938
RDY   = 000200           654*
RD.ADR  017676          4074*  4075*  4077*  4779
RD.RM   017652          3281   3300   3306   3529   3553   3567   3671   3685   3731   3754   3822   3832   3847
                        3981   4030   4072*  4127   4141   4147   4181
RD.RM1  017674          4076*  4106
RD.RM2  020020          4072*  4105*
RD.RM3  020024          4083   4097   4107*
RD.RM4  020030          4088   4109*
RD.WRD  017700          4078*  4079   4779   4781
READIN= 000121           828*
RECAL = 000107           823*
RELSE = 000113           825*
RESREG= 104413          2291   2849   2962*  3251   3384   3387   3451   3656   3716   3962   4190   4289
RESVEC= 000010           633*
RETRY   001250           908*  1569*  1593   1595*  1598*  1665   1667*  1670*
RMADR   012714          1282*  3159*  3231   3343   3469   3489   3510   3669   3710   3979   4074   4080   4084
                        4100   4133   4137   4170   4200   4953
RMAS  = 000016          3184*  3305*  3752   3871*  3898*  3916*  4021
RMBA  = 000004          2192   2194   3179*
RMCS1 = 000000          3177*  3273*  3279   3294   3420   3485   3506   3548   3568   3579   3594   3605   3732
                        3755   3807   3839   3876   4031   4128   4777
RMCS2 = 000010          1623   3181*  3232*  3272*  3274   3356*  3470*  3490*  3511*  3597   3616   3629   3641
                        3650*  3670*  3730*  3821*  3902*  3915*  3980*  3996*  4081   4138   4171*  4177   4201*
                        4206
RMDA  = 000006          1611   3180*  3477   3502   3517
RMDB  = 000022          3186*  4175
RMDC  = 000034          3191*  3481   3493   3524
RMDS  = 000012          1636   3182*  3301   3404   3406   3672   3823   3848   3903   3982
RMDT  = 000026          3188*  3282
RMEC1 = 000044          3195*
RMEC2 = 000046          3196*  4185
RMERRS  012546          2981*  3215   3790   3793   3803   3903*  3904*  3905*  3906*  4777
RMER1 = 000014          1625   1823   3183*  3307   3833   3904   4142
RMER2 = 000042          3194*  3905
RMHR  = 000036          3192*
RMINIT  012732          1289   1418   3211*
RMLA  = 000020          3185*  3686   3688
RMMR1 = 000024          3187*
RMMR2 = 000040          3193*  3906
RMOF  = 000032          3190*  3298   3530   3534   3554   3558
RMR   = 000004           724*
RMSN  = 000030          3189*
RMVEC   012716          1283*  3160*  3228   3230   3337
RMWC  = 000002          3178*
RM.REG  001306           931*   964   1611   1623   1625   1636   1823   2192   2194   4774   4779   4791   4795
                        4788   4791   4798   4801   4804   4808   4811   4814   4817
```

```
RM01    013436          1563    1578    1605    1693    3336#
RM01A   021265          1328    4393#
RM01B   021253          1322    4391#
RNOP  = 000101           821#
RP.MR   017124          1968    3946#
RP05    021260          1325    4392#
RTC   = 000117           827#
SAVEFG  012654          1290*   1419*   3110#   3583    3736    3895
SAVREG= 104412          2231    2823    2961#   3211    3339    3399    3615    3709    3949    4167    4273
SAVSEC  001252           909#   1611*   1613*   1614*   1615*   1641    1642    4794    4796    4808
SAVWC   001254           910#   1647*   1876
SC      015764          3715    3740    3742    3748    3752#
SCAWC   006074          1661    1876#
SC01  = 000100           781#
SC02  = 000200           782#
SC04  = 000400           783#
SC10  = 001000           784#
SC11    016616          3792    3829    3886#
SC12    016714          3786    3797    3902#
SC13    016764          3776    3779    3912#
SC20  = 002000           785#
SC3     016034          3767#   3771
SC4     016040          3769#   3800    3812    3816    3864    3878    3885    3901    3939
SC5     016052          3768    3774#
SC6     016252          3796    3817#
SC6A    016362          3794    3841#
SC7     016510          3831    3844    3846    3856    3866#
SC8     016566          3808    3824    3834    3840    3849    3877    3879#   3911
SEARCH= 000131           830#
SEC30   001264           914#   1376*   1382*   1559    1891
SEEK  = 000105           822#
SEEKFG  012656          1420*   3118#   3216    3432
SELDRV= 000145           833#
SETFMT= 000143           832#   1562
SETHDR  006122          1557    1684    1730    1862    1886#
SETPAT  004004          1505    1520    1526#
SETTBL  005640          1556    1683    1830#
SETVEC  002462          1280    1288#
SET.IE  020342          3245    3276    3374    3449    3655    3761    4199#
SKI   = 040000           807#
SLASH   021323          2021    4399#
SOFSW   001216           895#   1568*   1631    1651*   1653    1655    1657*   1662*   1879#
SRCHWT  012630          3064#   3430    3466*   3508*   3890    3899*
STACK = 001100           537#   1234    1577    1604    1692    1776    1973
STCLK1  006430          1942    1952#
STCLK2  006500          1953    1961#
STCLK3  006504          1951    1960    1962#
STKLMT= 177774           548#
ST0     017216          3956    3977#
ST01    017246          3986#   4029    4034
ST02    017444          3985    3987    3989    4021#
ST03    017514          4023    4035#
ST05    017540          3983    4002    4032    4041#
ST06    017546          4025    4043#
ST07    017604          4027    4051#
ST08    017634          4050    4057#
```

```
ST09    017644          4020    4038    4040    4042    4048    4055    4059#
ST.CLK  006352          1288    1417    1942#
SVRH70  020224          3585    3602    3633    3738    3745    3785    3836    3854    3897    3934    3995    4058    4167#
SWR     001140           869#   1232    1246*   1248    1254*   1273    1293    1671    1773    1816    2200    2207    2212
                        2216    2556    2595    2650*
SWREG   000176           514#   1254    1273    2556    2595    2618
SW0   = 000001           601#   1773
SW00  = 000001           591#    601
SW01  = 000002           590#    600
SW02  = 000004           589#    599    1293
SW03  = 000010           588#    598
SW04  = 000020           587#    597
SW05  = 000040           586#    596
SW06  = 000100           585#    595
SW07  = 000200           584#    594
SW08  = 000400           583#    593
SW09  = 001000           582#    592    1816
SW1   = 000002           600#
SW10  = 002000           581#
SW11  = 004000           580#
SW12  = 010000           579#
SW13  = 020000           578#
SW14  = 040000           577#
SW15  = 100000           576#
SW2   = 000004           599#
SW3   = 000010           598#
SW4   = 000020           597#
SW5   = 000040           596#
SW6   = 000100           595#
SW7   = 000200           594#
SW8   = 000400           593#
SW9   = 001000           592#
SYSTAT  021272          1298    4394#
T       021331          1997    4401#
TABLE   001376           974#   2012    2038
TAP   = 040000           776#
TBITVE= 000014           634#
TD      015624          3713    3722#
TIMER   012660          3123#   3320*   3444*   3588*   3634*   3725*   3818*   3862*   3913*   3920*   3952    3954*   4012*
                        4039*   4045*   4051*
TITLE   025732          1264    4869#
TKVEC = 000060           641#   1483*   2003*   2533*   2534*
TPVEC = 000064           642#
TRAPVE= 000034           640#   1238*   1239*   1259*
TRE   = 040000           659#
TRKCNT  001236           903#   1838*   1847    1849*   1853*
TRKTST  005730          1674    1845#
TRK1  = 004000           786#
TRK10 = 040000           789#
TRK2  = 010000           787#
TRK20 = 100000           790#
TRK4  = 020000           788#
TRNSWT  012626          1974    3056#   3467*   3612*   3622    3639*   3648*   3727    3728*   3990    4005    4018*
TRTVEC= 000014           635#
TTRKS   001232           901#   1502*   1503*   1504*   1515*   1516*   1517*   1518*   1521*   1839
TTRKSC  001234           902#   1839*   1845*
```

```
TYPADR  006604        1989#
TYPDS = 104405        1753    2020    2266    2954#
TYPE  = 104401        1264    1267    1297    1298    1305    1312    1314    1316    1318    1320    1321    1329    1331
                      1335    1342    1351    1353    1355    1357    1362    1371    1373    1379    1397    1408    1410
                      1425    1429    1431    1446    1455    1462    1472    1477    1487    1489    1496    1748    1750
                      1751    1754    1991    1992    1996    1997    2002    2015    2021    2033    2202    2210    2241
                      2242    2246    2247    2256    2269    2273    2278    2280    2286    2289    2328    2425    2501
                      2550    2562    2616    2617    2620    2631    2641    2652    2671    2719    2725    2730    2734
                      2739    2740    2742    2745    2749    2785    2950#   4920    4923    4934    4937
TYPERR  007512        2209    2231#
TYPOC = 104402        2262    2619    2951#   4922    4936
TYPON = 104404        2953#
TYPOS = 104403        1302    1493    2952#
ULDFLG  012634        3083#   3271#   3346    3361#   3545#   3608#   3640#   3782    3868    3870#   3886    3888#   3998#
                      4011#
UNS   = 040000         736#
UNTOFF  021170        1316    4379#
UNTON   021201        1320    4381#
UPDACY  006176        1734    1869    1904#
UPDATK  006322        1851    1931#
UPE   = 020000         683#
US1   = 000001         670#
US2   = 000002         671#
US4   = 000004         672#
VV    = 000100         709#
WC      001256         911#   1374#   1380#   1536
WCE   = 040000         684#   1623
WCF   = 000040         727#
WCKD  = 000151         834#
WCKHD = 000153         835#   1602
WCTBL   001600        1028#   1646
WLE   = 004000         733#   1823
WRAP    001304         926#   1386#   1448#   1457#   1506
WRL   = 004000         714#
WRTDAT= 000161         836#
WRTHD = 000163         837#   1575
WRTRK   004154        1556#
WRTRK1  004230        1568#   1673    1676
WRTRK2  004266        1575#   1652
WRT.AD  020122        4123#   4133#   4136#   4781
WRT.RM  020032        3293    3297    3419    3476    3480    3484    3492    3501    3505    3516    3523    3533    3547
                      3557    3578    3593    3604    3806    3838    3875    4120#
WRT.R1  020116        4134#   4154
WRT.R2  020206        4120#   4153#
WRT.R3  020214        4129    4140    4143    4149    4156#
WRT.R4  020220        4145    4158#
WRT.R5  020222        4157    4159#
WRT.WD  020120        4121#   4125    4132#   4135#   4781
$AUTOB  001134         866#   1277#   2609    2766
$BDADR  001122         861#
$BDDAT  001126         863#
$BELL   001162         880#   2202    2224    2562    2759
$CHARC  010156        2330#   2340#   2347    2355#   2361#
$CKSWR  011140        2595#   2958
$CMTAG  001100         849#   1229    1230    1240
$CM3  = 000000         879#
```

RMFOR.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$CNTLC  012035          2550    2631    2759*
$CNTLG  012047          2616    2761*
$CNTLU  012042          2641    2734    2760*
$CRLF   001167           882*   1297    1331    1335    1496    1754    2002    2210    2224    2241    2246    2256    2273
                        2286    2329    2364    2652    2739    2759
$DBLK   010624          2467    2501    2509*
$DB2D   012172          2804    2823*
$DECVL  012352          2825    2871*
$DOAGN  005442          1759    1764    1770*
$DTBL   010614          2470    2505*
$ENDAD  005432           522    1766*
$ENDCT  005416          1761*
$EOP    005336          1718    1745*   1825
$EOPCT  005410          1758*   1762
$ERFLG  001103           852*   2197*   2224
$ERMAX  001115           858*   1241*
$ERROR  007304          1236    2186*
$ERRPC  001116           859*   2204*   2205*   2206    2224    4785    4794    4796    4806    4808
$ERRTB  001700          1077*   2238
$ERTTL  001112           856*   1395*   1752    2203*   2224
$ESCAP  001160           879*   1240*   1584*   1597*   1616*   1633*   1635*   1669*   1699*   1709*   1819*   2219    2221
                        2224    3759    3810    3814    4090    4150    4359    4368    4369*   4371*
$FILLC  001156           877*   2333    2364
$FILLS  001155           876*   2364
$GDADR  001120           860*
$GDDAT  001124           862*   2196*   4808
$GET42  005422          1763*
$GTSWR  011230          2617*   2956
$HD   = 000000           493
$ICNT   001104           853*
$INTAG  001135           867*   2614*   2633    2653    2766
$ITEMB  001114           857*   2206*   2224    2233
$LF     001170           883*   2224    2364    2749    2759
$LKCSB  001200           890*   1949*
$LKCSR  001176           889*   1944    1950*
$LKS    001206           892*   1954    1959*
$LLVEC  001210           893*   1955
$LPADR  001106           854*
$LPERR  001110           855*   1576*   1603*   1691*   1818    2218
$LPVEC  001202           891*   1945
$MAIL = ****** U        1258    1273    2212    2317
$MNEW   012065          2620    2764*
$MSWR   012054          2617    2762*
$NULL   001154           875*   2335    2364
$OCNT   010404          2397*   2426*   2439*
$OMODE  010406          2392*   2396*   2401    2404*   2415*   2441*
$PASS   001100           850*   1755*   1756*   1773
$QUES   001166           881*   1351    1371    1410    1477    2224    2364    2671    2742    2759
$RDCHR  011502          2684*   2959
$RDDEC= ****** U        2961
$RDLIN  011572          2707*   2960
$RDOCT= ****** U        2961
$RDSZ = 000007          2700*
$RESRE  012424          2906*   2962
$RMADR  001172           887*   1282    4784    4919    4950    4952
$RMVEC  001174           888*   1283
```

RMFOR.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$RTNAD  005444          1772#
$R2A   = ******  U      2963
$SAVRE  012366          2890#   2961
$SB2D   012136          1994    2000    2802#
$SETUP= 000106          1224#   1235    1236    1238    1240    1242    1270    1755    2187    2215    2223    2554    2559
                        2560    2590    2766
$STUP  = 177777         1224#
$SUPRS  012076          1995    2001    2777#
$SVPC  = 000200          520#    525
$SWR   = 123000          483#    493     498     499     500     501     502     879     880    1240    1742    1755    1765
                        1771    1773    2178    2173    2180    2181    2182    2200    2207    2212    2216    2224
$TKB    001146           872#   1981    1985    2514    2535    2546    2571    2599    2626
$TKCNT  010634          2515#   2530*   2560    2577*   2691    2693*
$TKINT  010652          1269    1396    1988    2530#   2551    2612
$TKQEN= 010651          2519#   2585    2696
$TKQIN  010636          2516#   2531*   2532    2583*   2584*   2585    2587*
$TKQOU  010640          2517#   2532*   2694    2695*   2696    2698*
$TKQSR  010642          2518#   2531    2587    2698
$TKS    001144           871#   1986*   2514    2536*   2567*   2569    2575*   2597    2613*   2623    2635*   2655*
$TKSRV  010722          2533    2546#
$TN    = 000000          483#    493
$TNPWR  012302          2830    2831    2851#
$TPB    001152           874#   2353*   2364
$TPFLG  001157           978#   2311    2364
$TPS    001150           873#   2351    2364
$TRAP   012462          1238    2927#
$TRAP2  012504          2938#   2949
$TRP   = 000014          2942#   2951#   2952#   2953#   2954#   2955#   2956    2957#   2958    2959#   2960#   2961#   2962#
                        2963#
$TRPAD  012516          2932    2949#
$TSTNM  001102           851#   2199    2224
$TTYIN  012026          2709    2710    2722    2740    2754    2758#
$TYPBN= ******  U       2955
$TYPDS  010410          2455#   2954
$TYPE   007742          2311#   2942    2950
$TYPEC  010112          2332    2339    2346    2351#   2352    2657
$TYPEX  010160          2357    2359    2362#
$TYPOC  010206          2395#   2951
$TYPON  010222          2394    2397#   2953
$TYPOS  010162          2390#   2952
$$GET4= 000000          1765#
$OFILL  010405          2391#   2395#   2405    2440#
$40CAT= ******  U       2209
.      = 026610          508#    512#    520     521#    523#    525#    529#    848#    884    1233    1576    1603    1691
                        1773    2224    2364    2509#   2514    2518#   2519    2520#   2758#   2759    2766    2871#   4259#
                        4259#   4260#   4261#   4262#   4263#   4264#   4265#   4266
```

```
ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*RMFOR.BIN,RMFOR.SEQ/SOL/DOC/NL:MC:MD:CND/LI:ME/CRF:SYM/NL:TOC=RMFOR.P11
RUN-TIME:  67 53 4 SECONDS
RUN-TIME RATIO: 3284/125=26.3
```

CORE USED: 37K (73 PAGES)

DOCUMENT PAGES: 106