RM03

RM03 FORMATTER

MD-11-DZRMA-B

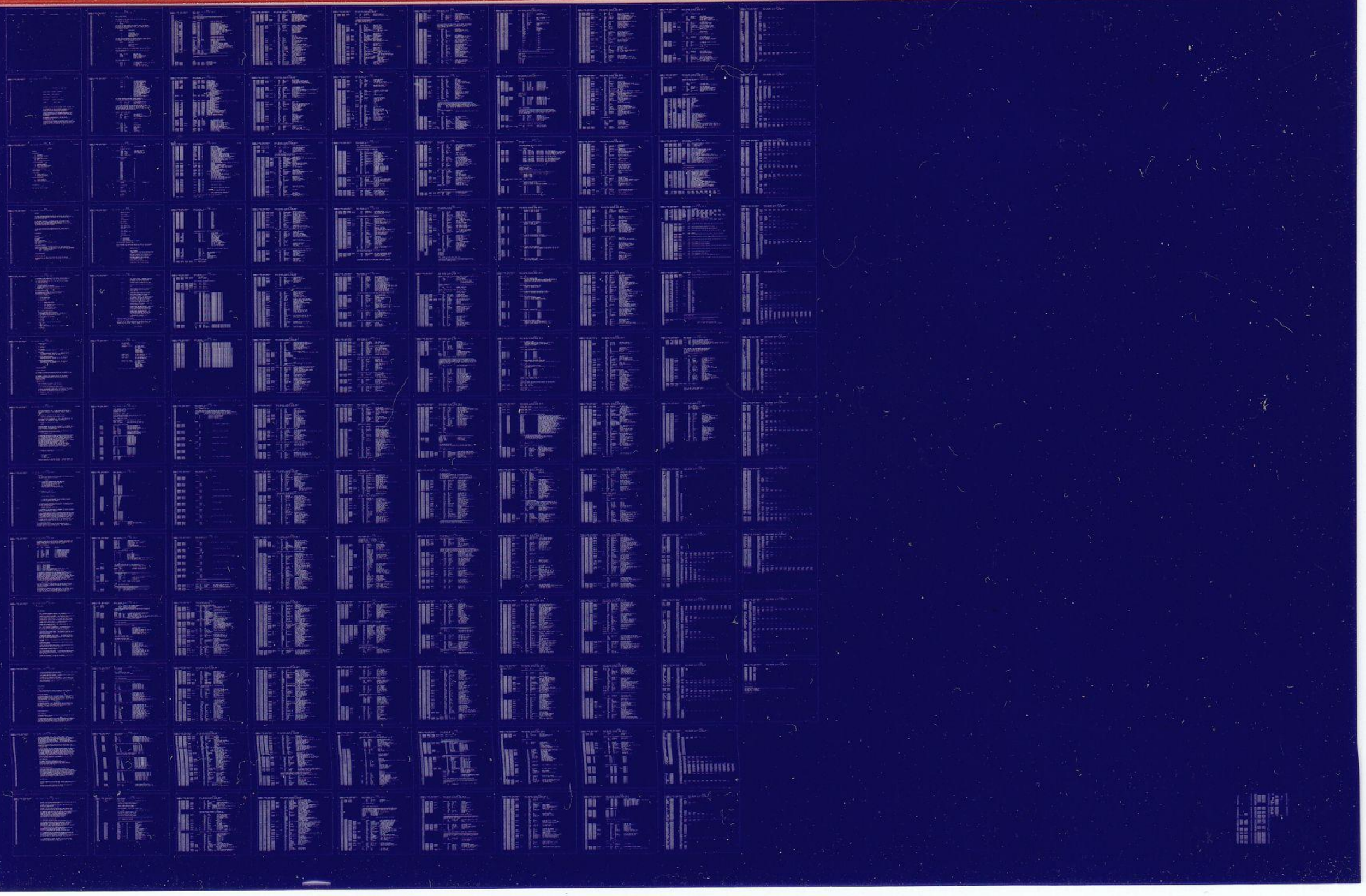
EP-DZRMA-B-DL-A

FICHE 1 OF 1

OCT 1977

digital

MADE IN USA



B01

EOF1?CMMLBSEQ	00010000	770804	PDP10 411	DOHDR1DZRMABSEQ	00010000	770804

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
```

.REM %

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

PRODUCT CODE: MAINDEC-11-DZRMA-B-D

PRODUCT NAME: RM03 FORMATTER

DATE CREATED: AUGUST 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. HESS/C. CHEN

CONTENTS

1. ABSTRACT

THE RM03 FORMATTER PROGRAM PROVIDES THE FACILITIES TO FORMAT OR
TO CHECK 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.
THE PROGRAM THEN VERIFIES THE WRITTEN DATA BLOCKS VIA EXECUTING THE
"WRITE CHECK HEAD AND DATA" COMMAND.

IN THE CHECK OPERATION,THE PROGRAM REPEATS THE FORMAT OPERATION
THREE TIMES WITH DATA PATTERN BEING ROTATED ONE BIT AT EACH
PASS.

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 RM03 DISK DRIVES
DISK PACK(S)

2.2 PRELIMINARY PROGRAMS

THERE ARE NO PREQUISITE PROGRAMS PROVIDING THE RM03 SUBSYSTEM IS
KNOWN TO BE OPERATIONAL. IF THE STATE OF THE RM03 SUBSYSTEM IS UNKNOWN,
THE FOLLOWING PROGRAMS SHOULD BE RUN PRIO TO USING THE FORMATTER PROGRAM

RM03	DISKLESS DIAGNOSTIC
RM03	FUNCTIONAL TEST

3. LOADING PROCEDURES

3.1 PAPER TAPE AND XXDP

THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE
LOADER OR IT MAY BE LOADED FROM 'XXDP' MEDIA USING THE APPROPRIATE
LOADER.

3.2 APT

```
                        THIS PROGRAM IS APT COMPATIBLE TO THE EXTENT THAT APT HOOKS WILL
                        BE IN THE PROGRAM AND WILL WORK THRU THE 'OPTION INTERFACE'.

                        FOR OTHER INTERFACES,APT MAY ONLY LOAD AND START THE PROGRAM.
                        I.E. LOAD AND DUMP MODE.

                        AUTOMATIC MODE (MONITOR)

                        1.  THE INPUT DIALOGUE IS BYPASSED.
                        2.  THE BUSS ADDRESS AND CONTROLLER INTERRUPT VECTOR IS
                            DEFAULTED.
                        DUMP MODE: INPUT DIALOGUE AFTER PROGRAM STARTS

                3.3     APT ETABLE DEFINITIONS

                        THE FOLLOWING DEFINITIONS ARE VALID FOR SPECIFYING APT
                        ENVIRONMENTAL TABLE (ETABLE) ENTRIES,VIA RUNNING
                        THE APT UTILITY PROGRAM "TSP":

                        1.   SOFTWARE ENVIRONMENT:

                             = 1 IF APT SCRIPT MODE
                             = 0 IF STANDLONE MODE

                        2.   ENVIRONMENT MODE:

                             BIT 7 = 1  ETABLE DOES SIZING
                                   = 0  PROGRAM DOES SIZING

                             BIT 6 = 1  SPOOL MESSAGES TO APT IF SCRIPT MODE
                                   = 0  DON'T SPOOL TO APT

                             BIT 5 = 1  SUPPRESS CONSOLE OUTPUT
                                   = 0  ALLOW CONSOLE OUTPUT

                             BIT 4 TO BIT 0 ARE NOT USED

                        3.   SWITCH 1 (SOFTWARE SWITCH REGISTER)
                             IF ENVIRONMENT MODE BIT 7 (SIZING BIT ) IS SET TO 1,
                             THE SOFTWARE SWITCH REGISTER WILL BE USED, INSTEAD
                             OF THE HARDWARE CONSOLE SWITCH REGISTER.

                        4.   SWITCH 2 (USER SWITCH REGISTER)
                                NOT USED

                        5.   CPU OPTIONS
                                NOT USED

                        6.   MEMORY TYPES 1-4 AND MAX MEMORY ADDRESSES
                                NOT USED

                        7.   INTERRUPT VECTOR 1:
                                USED WHEN ENVIRONMENT MODE BIT 7 = 1;DEFAULT = 254

                        8.   BUS PRIORITY 1:
                                NOT USED.
```

9. INTERRUPT VECTOR 2:
NOT USED

10. BUS PRIORITY 2:
NOT USED

11. BASE ADDRESS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1;DEFAULT = 176700

12. DEVICE MAP:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. EACH BIT SET TO 1 IN BITS 0 TO 7 WILL SELECT THE CORRESPONDING DRIVE TO BE TESTED. BITS 8-15 ARE NOT USED.

13. CONTROLLER DESCRIPTOR WORDS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. ANY NONE ZERO NUMBER SPECIFIES THAT PROGRAM RUNS IN CHECK MODE.

14. CONTROLLER DESCRIPTOR WORDS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. ANY NONE ZERO NUMBER SPECIFIES THAT PROGRAM SELECTS 30 SECTORS FOR A TRACK IN ALL OPERATIONS.

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:
                            (IN CASE OF FORMAT MODE)



                            'SELECT DATA PATTERN
                              (0) ZERO'S
                              (1) ONE'S
                              (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:

066666

NOTE: IN THE CHECK OPERATION, THE ABOVE MENTIONED MESSAGE IS BYPASSED. THE DATA PATTERN IS INITIATED TO 133331 AND IS ROTATED ONE BIT TO THE RIGHT AT EACH PASS OF THE CHECK OPERATION. AFTER THE CHECK OPERATION IS DONE, THE DATA PATTERN WILL BE RESTORED TO 066666.

9. THE PROGRAM WILL THEN TYPE:

'STARTING FORMAT ON DRIVE N'

OR

'STARTING CHECK ON DRIVE N'

IF A NEW PACK IS UNDER FORMATTED, THE PROGRAM ALSO ASKS TO ENTER TWO SERIAL NUMBERS. THESE TWO SERIAL NUMBER MUST BE NON-ZERO DECIMAL NUMBERS.

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 RH70
ADDRESS IN RESPONSE TO THE REQUEST FROM THE PROGRAM.

4.4 OTHER UNIBUS ADDRESSES

LOC	TAG	CONTENTS	FUNCTION
1160	$TKS	177560	TTY KEYBOARD STATUS REGISTER
1162	$TKB	177562	TTY KEYBOARD BUFFER REGISTER
1164	$TPS	177564	TTY PRINTER STATUS REGISTER
1166	$TPB	177566	TTY PRINTER BUFFER REGISTER
1276	$LKCSR	172540	KW11-P CONTROL REGISTER
1300	$LKCSB	172542	KW11-P COUNTER REGISTER
1302	$LPVEC	104	KW11-P VECTOR ADDRESS
1304	$LKS	177546	KW11-L CONTROL REGISTER
1306	$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 RM03 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 10 MINUTES TO FORMAT AN ENTIRE RM03 PACK. THE 'CHECK' MODE TIME FOR AN ENTIRE RM03 PACK IS 24 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
THE FORMAT OPERATION DESCRIBED IN SECTION 8.1,
EXCEPT THAT IN EACH CHECK OPERATION THE FORMAT OPERATION
IS REPEATED THREE TIMES WITH DATA PATTERN BEING ROTATED
ONE BIT POSITION AT EACH PASS.

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.

9. BAD SPOT FILE

SECTORS 0 THROUGH 31,ON CYLINDER 822.,TRACK 4 ARE ALWAYS FORMATTED
IN 16 BIT MODE. THEY CONTAIN THE BAD SECTOR ADDRESS ON THE
PACK.

SECTORS 0,2,4,6,8 ARE IDENTICAL AND RECORD THE MANUFACTURE
DEFINED BAD SECTOR ADDRESS FOR 16 BIT MODE.

```
598
599                     SECTORS 1,3,5,7,9 ARE IDENTICAL AND RECORD THE MANUFACTURE DEFINED
600                     BAD SECTOR ADDRESS FOR 18 BIT MODE.
601
602                     SECTORS 10,12,14,16,18,20,22,24,26,28,30 ARE IDENTICAL AND
603                     RECORD THE  USER DEFINED
604                     BAD SECTOR ADDRESS FOR 16 BIT MODE.
605
606                     SECTORS 11,13,15,17,19,21,23,25,27,29,31 ARE IDENTICAL AND
607                     RECORD THE USER DEFINED BAD SECTOR ADDRESS FOR 18 BIT MODE.
608
609                     THE FORMAT OF EVERY SECTOR OF THE BAD SPOT FILE IS DESCRIPTED BELOW:
610                     THE FIRST TWO WORDS IN THE DATA FIELD SPECIFY THE SERIAL
611                     NUMBER OF THE PACK.  THE SERIAL NUMBER MUST BE A NON-ZERO
612                     NUMBER AND CONSISTS OF MAXIMUM 10 OCTAL DIGITS.
613                     THE FIRST WORD OF THE SERIAL NUMBER CONTAINS THE 5
614                     LEST SIGNIFICANT OCTAL DIGITS,WHILE THE SECOND
615                     WORD CONTAINS THE 5 MOST SIGNIFICANT DIGITS.
616
617                     THE THIRD WORD IS ALWAYS ZERO.
618
619                     THE FORTH WORD DEFINES THE PACK IS A DATA PACK ,IF IT IS 0.
620
621                     THE FOLLOWING 252 WORDS DEFINE THE BAD SECTOR ADDRESS;TWO
622                     WORDS ARE USED TO DEFINE A SIGNLE BAD SECTOR, THE FIRST
623                     WORD SPECIFIES THE CYLINDER ADDRESS, THE SECOND WORD
624                     SPEDIFIES THE TRACK (HIGH BYTE) AND SECTOR (LOW BATE) ADDRESSES.
625                     ALL UNUSED LOCATIONS ARE RECORDED WITH  ONES.
626
627                     FOR A UNFORMATTED PACK,ALL MANUFACTURE DEFINED SECOTRS ARE
628                     RECORDED WITH NO BAD SECTORS,ALL BAD SECTORS
629                     DETECTED BY THE PROGRAM ARE RECORDED IN SECTORS 10,12,... OR
630                     11,13,... DEPENDED ON THE 16 BIT MODE OR 18 BIT MODE.
631
632                     FOR A FORMATTED PACK,ALL MANUFACTURE DEFINED BAD SECTORS ARE
633                     NOT REFORMATTED OR NOT WRITTEN ANY DATA WORDS BY THE PROGRAM.
634                     BESIDES,FOR A FORMATTED PACK,IF THE PROGRAM RUNS
635                     IN 16 BITS MODE, ALL MANUFACTURE DEFINED AND 18 BIT MODE USER DEFINED
636                     BAD SECTORS ARE NOT UPDATED OR DESTORIED ,AND VICE VERSA.
637
638
639                     IT IS RECOMMENDED TO FORMAT THE PACK IN 18 BIT MODE FIRST
640                     THEN RESTART THE PROGRAM TO FORMAT THE PACK IN 16 BIT MODE
641                     IN CASE A UNFORMATTED PACK IS SELECTED.
642
643
644
645
```

10.1 RH70(11)/RM03 DRIVER

THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH70/RM03 DRIVER.

10.2 TO INITIALIZE THE DRIVER:

```
        JSR     PC,RMINIT
        RETURN
```

UPON RETURN YOU MUST EXAMINE THE "DRVSTA" TABLE TO DETERMINE THE DRIVES THAT ARE ONLINE FOR TESTING. THE 'DRVSTA' TABLE IS EIGHT BYTES; ONE BYTE PER DRIVE. THE STATE OF EACH DRIVE WILL BE INDICATED AS FOLLOWS:

DRVSTA	DRIVE STATE
>0	ONLINE RM03
=0	OFFLINE RM03, DRIVE IS NOT AN RM03, OR NONEXISTENT DRIVE
<0	UNSAFE RM03

THE DRIVE TYPE IS DEFINED IN AN 8 BYTE LONG TABLE TAGGED 'DRVTYP'. THE TABLE CONTAINS ONE BYTE FOR EACH DRIVE AND IS INDEXED BY THE DRIVE NUMBER. ENTRIES ARE ENCODED AS FOLLOWS:

DRVTYP	CONDITION
0	NONEXISTENT DRIVE
4	RM03
-1	NOT AN RM03

THE 'RMINIT' ROUTINE WILL DO A READIN PRESET AND WILL SET FMT22.

10.3 AFTER THE DRIVER HAS BEEN INITIALIZED, IT IS CALLED USING THE FOLLOWING SEQUENCE.

CALL:

```
        JSR     R0,RM03         ;MAKE THE CALL
        PNTDPB                  ;ADDRESS OF DPB*
        RETURN1                 ;RETURN IF QUEUE IS FULL
        RETURN2                 ;RETURN IF REQUEST IS IN
                                ;QUEUE OR THERE IS AN
                                ;ERROR CONDITION
```

*DPB (DATA PARAMETER BLOCK)

```
PNTDPB: .BYTE   0               ;(0) DRIVE NUMBER
        .BYTE   0               ;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
        .BYTE   0               ;(2) COMMAND
        .BYTE   0               ;(3) PSEL AND A17 AND A16
        .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
```

```
                        .WORD   0               ;(6) BUFFER ADDRESS OR
                                                ;REGISTER TABLE POINTER
                        .BYTE   0               ;(10) SECTOR ADDRESS OR
                                                ;FIRST REG. INDEX
                        .BYTE   0               ;(11) TRACK ADDRESS OR
                                                ;LAST REG. INDEX
                        .WORD   0               ;(12) CYLINDER ADDRESS
                        .WORD   0               ;(14) ERROR TABLE POINTER
                                                ;POINTS TO THE FIRST OF TWENTY
                                                ;LOCATIONS OF WHERE THE DRIVER
                                                ;IS TO STORE THE RH70/RM03
                                                ;REGISTERS ON AN ERROR. IF LEFT
                                                ;ZERO REGISTERS ARE NOT SAVED.
                        .WORD   0               ;(16) STATUS/ERROR INDICATOR
                                                ;BIT15=1=>ERROR OCCURRED
                                                ;BIT07=1=>DONE
                                                ;BIT14-BIT09 AND BIT06-BIT03
                                                ;INDICATE TYPE OF ERROR

10.4    THE DRIVER PROVIDES A SOFTWARE TIMEOUT CAPABILITY.
        TO UTILIZE THIS CAPABILITY YOU MUST SUPPLY THE "RM TIMER" ROUTINE
        WITH THE ELAPSED TIME IN THE FOLLOWING MANNER:

                MOV     #16.,-(SP)      ;16 MILLISECONDS BETWEEN
                                        ;CLOCK TICKS
                JSR     PC,RMTMR        ;CALL THE TIMER ROUTINE

        IT SHOULD BE NOTED THAT YOU MUST PROVIDE THE CODE TO DRIVE THE
        CLOCK.  AND THE ELAPSED TIME MUST BE IN MILLISECONDS.
        THE DRIVER WILL SET THE TIMEOUT TO 1 SECOND FOR ALL POSITIONING
        AND DATA TRANSFER OPERATIONS AND WILL SET THE TIMOUT TO 30
        SECONDS FOR ERROR RECOVERY OPERATIONS.


10.4.1  EXAMPLE - WRITE 1000. WORDS

        1$:     JSR     R0,RM03         ;CALL THE DRIVER
                WRTDPB                  ;DPB ADDRESS
                BR      1$              ;WAIT FOR QUEUE IF FULL
        2$:     TST     WRTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
                BEQ     2$
                BMI     ERROR1          ;ERROR OCCURRED
                .
                .
                .

        WRTDPB: .BYTE   5               ;DRIVE #5
                .BYTE   0               ;
                .BYTE   161             ;WRITE COMMAND
                .BYTE   0               ;
                .WORD   -1000.          ;WORD COUNT
                .WORD   WRTBUF          ;BUFFER ADDRESS
                .BYTE   3               ;SECTOR
                .BYTE   5               ;TRACK
                .WORD   400             ;CYLINDER
                .WORD   ERRTB5          ;ERROR TABLE
```

```
                        .WORD   0               ;STATUS/ERROR INDICATOR

                        ALTERNATE DPB SETUP

                WRTDPB: .WORD   5               ;THIS SETUP ACHIEVED
                        .WORD   WRITE           ;EVERYTHING THE
                        .WORD   -1000.          ;ABOVE TABLE DID, BUT
                        .WORD   WRTBUF          ;IN A CLEANER FORMAT
                        .BYTE   3,5
                        .WORD   400,ERRTBS,0

10.5    RH70/RM03 REGISTERS

                        MNEMONIC                INDEX
                        --------                -----

                        RMCS1                     0
                        RMWC                      2
                        RMBA                      4
                        RMDA                      6
                        RMCS2                     10
                        RMDS                      12
                        RMER1                     14
                        RMAS                      16
                        RMLA                      20
                        RMDB                      22
                        RMMR1                     24
                        RMDT                      26
                        RMSN                      30
                        RMOF                      32
                        RMDC                      34
                        RMHR                      36
                        RMMR2                     40
                        RMER2                     42
                        RMEC1                     44
                        RMEC2                     46

10.6    COMMANDS PERFORMED BY THE DRIVER

                        COMMAND                 CODE            COMMAND TYPE
                        -------                 ----            ------------

                        NO OPERATION            101                 N

                        UNLOAD                  103                 N

                        SEEK                    105                 P

                        RECALIRATE              107                 P

                        DRIVE CLEAR             111                 N

                        RELEASE                 113                 N

                        OFFSET                  115                 P
```

RETURN TO CENTER	117	P
READIN PRESET	121	N
PACK ACKNOWLEDGE	123	N
SEARCH	131	P
GET REGISTER(S)	141	S
SET FORMAT	143	S
SELECT DRIVE	145	S
WRITE CHECK DATA	151	D
WRITE CHECK HEADER AND DATA	153	D
WRITE DATA	161	D
WRITE HEADER & DATA	163	D
READ DATA	171	D
READ HEADER & DATA	173	D

N = HOUSEKEEPING
P = POSITIONING
D = DATA TRANSFER
S = SPECIAL PROVIDED BY THE DRIVER

10.7 DPB STATUS/ERROR INDICATOR WORD

THIS INDICATOR WILL INFORM THE USER OF THE RESULTS OF THE REQUEST. THIS IS ACCOMPLISHED BY SETTING VARIES BITS OF THE INDICATOR TO A ONE.

BIT NO.	MEANING IF ON A "1"
15	ERROR OCCURRED DONE (BIT07=0); BITS 14-10 SPECIFIES TYPE DONE (BIT07=1); BITS 06-03 SPECIFIES TYPE
14(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON AN OFFLINE OR UNSAFE DRIVE
13(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON A DRIVE THAT HAS AN UNLOAD REQUEST IN QUEUE.
12(2)	PERSISTENT UNSAFE CONDITION EXIST.
11(2)	UNCORRECTABLE PARITY ERROR OCCURRED

10(2)(4)	FATAL PARITY ERROR. A MASSBUS CLEAR WAS PERFORMED, ALL QUEUES WERE EMPTIED, AND ALL DRVACT'S SET TO THE IDLE STATE
9(3)(4)	SOFTWARE TIMEOUT OCCURRED ON THIS DRIVE
8(4)	SOFTWARE TIMEOUT OCCURRED ON ANOTHER DRIVE
7	DONE
6(2)	ERROR OCCURRED DURING AN I/O OPERATION
5(2)	ERROR OCCURRED DURING AN OPERATION OTHER THAN I/O.
4(2)	CORRECTABLE UNSAFE CONDITION OCCURRED
3(2)	DRIVE ERROR OCCURRED THAT CAUSED AN AUTOMATIC "RECALIBRATE" SEQUENCE
2	PORT REQUEST TIMEOUT. THE DRIVER REQUESTED THE DRIVE BUT THE OPPOSITE PORT DID NOT RELEASE THE DRIVE WITHIN 20 SECONDS.
1	NON-EXISTENT DRIVE REQUESTED. USER MADE A REQUEST FOR A NON-EXISTENT DRIVE.
(1) =>	REQUEST WASN'T PUT IN QUEUE. (RH70/RM03 REGISTERS WERE NOT SAVED)
(2) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A "DRIVE CLEAR" TO THE DRIVE. NOTE: ALL RH70/RM03 REGISTERS ARE SAVED AS PER DPB+14 BEFORE THE "DRIVE CLEAR".
(3) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A MASSBUS INIT. ALL RH70/RM03 REGISTERS FOR THE DRIVE WERE SAVED AS PER DPB+14 BEFORE THE INIT.
(4) =>	A "RECALIBRATE" SHOULD BE ISSUED BEFORE ANY OTHER COMMAND.

10.8 ERROR CALLS MADE BY THE DRIVER.

THERE ARE A FEW ERRORS THAT CAN OCCUR THAT CAN NOT BE INDICATED IN A DPB.

WHEN THIS TYPE OF ERROR IS DETECTED BY THE DRIVER IT WILL MAKE AN ERROR CALL OF THE FORM "ERROR N", WHERE "N" IS THE ERROR NUMBER AND THE ERROR WILL BE AN EMT INSTRUCTION.

N	TYPE	DATA AVAILABLE

```
                        1       RH70 INTERRUPT          *R4= RMCS1'S ADDRESS
                                OCCURRED (RHAS=0)

                        2       UNEXPECTED ATTENTION     R1= DRIVE NUMBER
                                OCCURRED                 R3= ATA BIT
                                                        *R4= RMCS1'S ADDRESS
                                                         R5= (RMAS)
                                                         RMERRS  =RMDS
                                                         RMERRS+2=RMER1
                                                         RMERRS+4=RMER2
                                                         RMERRS+6=RMMR2

                        3       MASSBUS PARITY           RD.ADR= ADDRESS OF REG. READ
                                ERROR (MCPE=1)           RD.WRD= WORD READ

                        4       MASSBUS PARITY           WRT.AD= ADDRESS OF REG. WRITTEN
                                ERROR (PAR=1)            WRT.WD= WORD WRITTEN
                                                         RD.WRD= WORD READ BACK

                        5       ADDRESS PLUG CHANGE      R1= DRIVE NUMBER
                                BIT SET ('OPE' ERROR)    R3= ATA BIT
                                                        *R4= RMCS1'S ADDRESS
                                                         R5= (RMAS)
                                                         RMERRS  =RMDS
                                                         RMERRS+2=RMER1
                                                         RMERRS+4=RMER2
                                                         RMERRS+6=RMMR2

                        * THIS IS THE ACTUAL UNIBUS ADDRESS (176700)

                        %
```

```
                        .TITLE  MAINDEC-11-DZRMA, RM03 FORMATTER
                        ;*COPYRIGHT (C) 1977
                        ;*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-C3), JAN 19, 1977.
                        ;*
                        .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
```

```
                                ;;BREAKPOINT TRAP (BPT)
                                ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                                ;;POWER FAIL
                                ;;EMULATOR TRAP (EMT) **ERROR**
                                ;;"TRAP" TRAP
                                ;;TTY KEYBOARD VECTOR
                                ;;TTY PRINTER VECTOR
                                ;;PROGRAM INTERRUPT REQUEST VECTOR

                        .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

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

                        .SBTTL  ACT11 HOOKS

                        ;;**************************************************************
                        ;HOOKS REQUIRED BY ACT11
                                $SVPC=.                 ;SAVE PC
                                .=46
000046  015066                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                                .=52
000052  020000                  .WORD   20000           ;;2)SET LOC.52 TO 20000
                                .=$SVPC                 ;; RESTORE PC
                        .=200
000200  000137  006422          JMP     BEGIN1  ;NORMAT STARTING ADDRESS
000204  000137  006412          JMP     BEGIN   ;CHANGE THE RH11 ADDRESS

                        .=1100
                        .SBTTL  APT PARAMETER BLOCK

                        ;;**************************************************************
                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                        ;;**************************************************************
                                .$X=.   ;;SAVE CURRENT LOCATION
                                .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                  200     ;;FOR APT START UP
```

```
                    000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
                    001100                $APTHDR ;;POINT TO APT HEADER BLOCK
                    001100                .=.$X   ;;RESET LOCATION COUNTER
                                        ;;******************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.

                                        $APTHD:
001100                                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001100  000000                          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001102  001206                          $TSTM:  .WORD   200.    ;;RUN TIM OF LONGEST TEST
001104  000310                          $PASTM: .WORD   200.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001106  000310                          $UNITM: .WORD   200.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001110  000310                                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
001112  000032                          TAB.XY  =       .       ;CMTAGSTARTING ADDRESS
                    001114

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

                                        .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  RM03 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          DTE=    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)
```

```
                                ;DIAGINOSTIC MODE (BIT #0)
                                ;MAINTAINABILITY CLOCK (BIT #1)
                                ;MAINTAINABILITY INDEX (BIT #2)
                                ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
                                ;MAINTAINABILITY READ (BIT #4)
                                ;MAINTAINABILITY WRITE (BIT #5)
                                ;MAINTAINABILITY SYNC DETECTED (BIT #7)

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

                                ;DEVICE 0 (BIT #0)
                                ;DEVICE 1 (BIT #1)
                                ;DEVICE 2 (BIT #2)
                                ;DEVICE 3 (BIT #3)
                                ;DEVICE 4 (BIT #4)
                                ;DEVICE 5 (BIT #5)
                                ;DEVICE 6 (BIT #6)
                                ;DEVICE 7 (BIT #7)

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

                                ;DRIVE TYPE REGISTER (RMDT) (#06)

                                ;DRIVE TYPE NUMBER BIT 1
                                ;DRIVE TYPE NUMBER BIT 2
                                ;DRIVE TYPE NUMBER BIT 3
                                ;DRIVE TYPE NUMBER BIT 4
                                ;DRIVE TYPE NUMBER BIT 5
                                ;DRIVE TYPE NUMBER BIT 6
                                ;DRIVE TYPE NUMBER BIT 7
                                ;DRIVE TYPE NUMBER BIT 8
                                ;DRIVE TYPE NUMBER BIT 9
                                ;DRIVE REQUEST REQUIRED (BIT #11)
                                ;MOVING HEAD (BIT #13)
                                ;TAPE DRIVE (BIT #14)
                                ;NOT BLOCK ADDRESSED (BIT #15)

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

                                ;SECTOR COUNT FIELD 0 (BIT #6)
                                ;SECTOR COUNT FIELD 1 (BIT #7)
                                ;SECTOR COUNT FIELD 2 (BIT #8)
                                ;SECTOR COUNT FIELD 3 (BIT #9)
                                ;SECTOR COUNT FIELD 4 (BIT #10)
                                ;TRACK FIELD 1 (BIT #11)
                                ;TRACK FIELD 2 (BIT #12)
                                ;TRACK FIELD 3 (BIT #13)
                                ;TRACK FIELD 4 (BIT #14)
                                ;TRACK FIELD 5 (BIT #15)

                                ;OFFSET REGISTER (RMOF) (#11)

                                ;OFFSET DIRECTION FOR RM03
                                ;HEADER COMPARE INHIBIT (BIT #10)
                                ;ERROR CORRECTION CODE INHIBIT (BIT #11)
```

```
1294            010000          FMT16=  10000                   ;FORMAT BIT (BIT #12
1295
1296                            ;DESIRED CYLINDER ADDRESS (RMDC) (#12)
1297                            ;(EACH BIT IS CALLED BY BIT NUMBER)
1298
1299                            ;SERIAL NUMBER REGISTER (RMSN) (#14)
1300                            ;(EACH IS CALLED BY BIT NUMBER)
1301
1302                            ;RM03 MAINTENANCE  REGISTER #02 (RMMR2) (#15)
1303
1304            040000          SKI=    40000                   ;SEEK INCOMPLETE (BIT #14)
1305
1306                            ;ECC POSITION REGISTER (RMEC1) (#16)
1307                            ;(EACH BIT IS CALLED BY BIT NUMBER)
1308
1309                            ;ECC PATTERN REGISTER (RMEC2) (#17)
1310                            ;(EACH BIT IS CALLED BY BIT NUMBER)
1311
1312                            ;;*****************************************************************
1313
1314                            .SBTTL  RM03 DRIVER COMMANDS
1315
1316                            ;;*****************************************************************
1317
1318            000101          RNOP    =       101             ;NO OPERATION
1319            000105          SEEK    =       105             ;SEEK
1320            000107          RECAL   =       107             ;RECALIBRATE
1321            000111          DRVCLR  =       111             ;DRIVE CLEAR
1322            000113          RELSE   =       113             ;RELEASE
1323            000115          OFFSET  =       115             ;OFFSET
1324            000117          RTC     =       117             ;RETURN TO CENTER LINE
1325            000121          READIN  =       121             ;READ IN PRESET
1326            000123          ACK     =       123             ;PACK ACKNOWLEDGE
1327            000131          SEARCH  =       131             ;SEARCH
1328            000141          GETREG  =       141             ;GET REGISTERS
1329            000143          SETFMT  =       143             ;SET FORMAT (& ECI OR HCI)
1330            000145          SELDRV  =       145             ;SELECT DRIVE
1331            000151          WCKD    =       151             ;WRITE CHECK DATA
1332            000153          WCKHD   =       153             ;WRITE CHECK HEADER & DATA
1333            000161          WRTDAT  =       161             ;WRITE DATA
1334            000163          WRTHD   =       163             ;WRITE HEADER & DATA
1335            000171          RDDAT   =       171             ;READ DATA
1336            000173          RDHD    =       173             ;READ HEADER & DATA
1337
1338
```

```
                                .SBTTL  COMMON TAGS

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

001114                                  .=TAB.XY
001114                          $CMTAG:                 ;;START OF COMMON TAGS
001114  000000                          .WORD   0
001116  000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001117  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001120  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001122  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001124  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001126  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001130  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001131  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001132  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001134  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001136  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001140  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001142  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001144  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001146  000000                          .WORD   0
001150  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001151  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001152  000000                          .WORD   0
001154  177570                  $SWR:   .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001156  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
001160  177560                  $TKS:   177560          ;;TTY KBD STATUS
001162  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001164  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001166  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001170  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001171  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001172  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001173  000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001174  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001176  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001202  077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
001203  015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001204  000012                  $LF:    .ASCIZ  <12>    ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .NLIST ME
                                ;
                                ;

                                .EVEN
001206                          $MAIL:                  ;;APT MAILBOX
001206  000000                  $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001210  000000                  $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001212  000000                  $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001214  000000                  $PASS:  .WORD   APASS   ;;PASS COUNT
```

```
1395  001216  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1396  001220  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1397  001222  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1398  001224  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1399  001226                          $ETABLE:                ;;APT ENVIRONMENT TABLE
1400  001226      000                 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1401  001227      000                 $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1402  001230  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1403  001232  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1404  001234  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1405                                  ;*                      BITS 15-11=CPU TYPE
1406                                  ;*                           11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1407                                  ;*                           11/70=06,PDQ=07,Q=10
1408                                  ;*                      BIT 10=REAL TIME CLOCK
1409                                  ;*                      BIT  9=FLOATING POINT PROCESSOR
1410                                  ;*                      BIT  8=MEMORY MANAGEMENT
1411  001236      000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1412  001237      000                 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1413                                  ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
1414                                  ;*                           900 NSEC CORE=001
1415                                  ;*                           300 NSEC BIPOLAR=002
1416                                  ;*                           500 NSEC MOS=003
1417  001240  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1418                                  ;*                       MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1419  001242      000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1420  001243      000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1421  001244  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1422  001246      000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1423  001247      000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1424  001250  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1425  001252      000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1426  001253      000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1427  001254  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1428  001256  000254                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1429  001260  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1430  001262  176700                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1431  001264  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1432  001266  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1433  001270  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1434  001272                          $ETEND:
1435                                  .MEXIT
1436          000015                  CR      =       15
1437          000012                  LF      =       12
1438  001272  176700                  $RMADR: .WORD   176700          ;RH11/RM03 UNIBUS ADDRESS
1439  001274  000254                  $RMVEC: .WORD   254             ;RH11 INTERRUPT VECTOR
1440  001276  172540                  $LKCSR: .WORD   172540          ;ADDRESS OF KW11-P CSR
1441  001300  172542                  $LKCSB: .WORD   172542          ;ADDRESS OF KW11-P COUNTER BUFFER
1442  001302  000104  000106          $LPVEC: .WORD   104,106         ;ADDRESS OF KW11-P VECTOR
1443  001306  177546                  $LKS:   .WORD   177546          ;ADDRESS OF KW11-L CONTROL REGISTER
1444  001310  000100  000102          $LLVEC: .WORD   100,102         ;ADDRESS OF KW11-L VECTOR
1445          001220                  DRIVE   =$UNIT                  ;CONTAINS DRIVE NUMBER SELECTED
1446                                                                  ;SAME PARAMETER USED IN APT
1447  001314  000000                  SOFSW:  .WORD   0               ;CONTENTS ARE FOR SOFTWARE DECISIONS
1448  001316  000000                  MODE:   .WORD   0               ;'FORMAT & VERIFY' OR 'CHECK' MODE INDICATOR
1449  001320  001466                  ENDCYL: .WORD   822.            ;ENDING CYLINDER
1450  001322  000000                  BEGCYL: .WORD   0               ;STARTING CYLINDER
```

```
1451  001324  000004  ENDTRK: .WORD   4.              ;ENDING TRACK
1452  001326  000000  BEGTRK: .WORD   0               ;STARTING TRACK
1453  001330  000000  TTRKS:  .WORD   0               ;TOTAL # OF TRACKS TO BE FORMATTED
1454  001332  000000  TTRKSC: .WORD   0               ;TOTAL # OF TRACKS COUNTER
1455  001334  000000  TRKCNT: .WORD   0               ;COUNTS TRKS FROM 0-5 PER CYL
1456  001336  000000  PATSEL: .WORD   0               ;CONTAINS PATTERN SELECTED
1457  001340  000000  PATA:   .WORD   0               ;1ST WORD OF PATTERN
1458  001342  000000  PATB:   .WORD   0               ;2ND WORD OF PATTERN
1459  001344  000000  CYLCK:  .WORD   0               ;STORE CYLINDER ADDRESS FOR FORMAT VERIFICATION
1460  001346  000000  RETRY:  .WORD   0               ;MAINTAINS # OF WRITE RETRIES MADE
1461  001350  000000  SAVSEC: .WORD   0               ;CONTAINS LAST BAD SECTOR ON FORMAT
1462  001352  000000  SAVWC:  .WORD   0               ;CONTAINS WC FOR REMAINING SECTORS ON ERROR
1463  001354  000000  WC:     .WORD   0               ;30 OR 32 SECTOR TRACK SIZE (IN WORDS)
1464  001356  000000  MWC:    .WORD   0               ;2'S COMPLEMENT OF 'WC'
1465  001360  000000  HEDERR: .WORD   0               ;POSITIONING ERROR DURING FORMAT INDICATOR
1466  001362  000000  SEC30:  .WORD   0               ;30 OR 32 SECTOR MODE INDICATOR
1467                                                  ;0 = 30 SECTOR MODE
1468                                                  ;1'S = 32 SECTOR MODE
1469  001364  000000  MAXSEC: .WORD   0               ;MAXIMUM SECTOR ADDRESS (FOR EITHER 30 OR 32 SECTOR
1470                                                  ;FORMAT)
1471  001366  000000  DS.CYL: .WORD   0               ;ADDRESS OF CURRENT CYLINDER
1472  001370  000000  DS.TRK: .WORD   0               ;ADDRESS OF CURRENT TRACK
1473  001372  000000  DDRIVE: .WORD   0               ;DRIVE NUMBER OF 'DRIVER' ERROR MESSAGES
1474  001374  000000  ATTN:   .WORD   0               ;ATTENTION REGISTER IMAGE FOR 'DRIVER'
1475                                                  ;ERROR MESSAGES
1476  001376  000000  CNTLC:  .WORD   0               ;ADDRESS OF '↑C' RETURN
1477  001400  000000  CHGADR: .WORD   0               ;'CHANGE RH11 ADDRESS' FLAG
1478  001402  000000  WRAP:   .WORD   0       ;WRAP AROUND FLAG,FORMAT FROM HIGH TO LOW
1479                                                  ;CYLINDER NUMBER
1480  001404  000000  DEVMP:  .WORD   0               ;DEVICE MAP FOR APT
1481  001406  000000  PACK:   .WORD   0       ;PACK>0, BAD SPOT FILE AVAILABLE FROM PACK
1482  001410  000000  EDIT:   .WORD   0               ;EDIT THE BAD SPOT FILE ,IF EDIT>0
1483  001412  000000  HEADX:  .WORD   0               ;HEAD BIT INDICATOR
1484                                                  ;BIT15!BIT14  GOOD SECTOR
1485                                                  ;BIT15 ALONE, USER DEFINED BAD SECTOR
1486                                                  ;BIT14 ALONE, MFG DEFINED BAD SECTOR
1487  001414  000400  BAD16:  .BLKW   256.            ;MFG 16 BIT BAD SPOT TABLE
1488  002414  177777          .WORD   -1
1489  002416  177777          .WORD   -1
1490  002420  000400  BAD18:  .BLKW   256.            ;MFG 18 BIT BAD SPOT TABLE
1491  003420  177777          .WORD   -1
1492  003422  177777          .WORD   -1
1493  003424  000400  USTAB:  .BLKW   256.            ;USER BAD SPOT TABLE
1494  004424  177777          .WORD   -1
1495  004426  177777          .WORD   -1
1496  004430  000400  USSAV:  .BLKW   256.            ;USER BAD SPOT TABLE SAVE FROM PACK
1497  005430  177777          .WORD   -1
1498  005432  177777          .WORD   -1
1499  005434  000040  SECCU:  .BLKW   32.      ;32 WORDS MAX TO STORE BAD SECTORS
1500  005534  177777          .WORD   -1
1501  005536  000000  NEXT:   .WORD   0               ;FLAG, IF NEXT=-1,ALL GOOD SPOT ON
1502                                                  ;THE CURRENT TRACK ARE FORMATTED
1503
1504                          ;RH11/RM03 REGISTERS STORED HERE AFTER AN OPERATION
1505
1506  005540  000000  RM.REG: .WORD   0               ;RMCS1
```

```
1507  005542  000000                          .WORD   0               ;RMWC
1508  005544  000000                          .WORD   0               ;RMBA
1509  005546  000000                          .WORD   0               ;RMDA
1510  005550  000000                          .WORD   0               ;RMCS2
1511  005552  000000                          .WORD   0               ;RMDS
1512  005554  000000                          .WORD   0               ;RMER1
1513  005556  000000                          .WORD   0               ;RMAS
1514  005560  000000                          .WORD   0               ;RMLA
1515  005562  000000                          .WORD   0               ;RMDB
1516  005564  000000                          .WORD   0               ;RMMR1
1517  005566  000000                          .WORD   0               ;RMDT
1518  005570  000000                          .WORD   0               ;RMSN
1519  005572  000000                          .WORD   0               ;RMOF
1520  005574  000000                          .WORD   0               ;RMCA
1521  005576  000000                          .WORD   0               ;RMDC
1522  005600  000000                          .WORD   0               ;RMER2
1523  005602  000000                          .WORD   0               ;RMMR2
1524  005604  000000                          .WORD   0               ;RMEC1
1525  005606  000000                          .WORD   0               ;RMEC2
1526
1527
1528                                  ;DATA/PARAMETER BLOCK - USED FOR ALL DRIVE OPERATION
1529
1530  005610      000                 FMTDPB: .BYTE   0               ;DRIVE NUMBER
1531  005611      000                         .BYTE   0               ;OFFSET VALUE OR FMT22,ECI, AND HCI
1532  005612      000                         .BYTE   0               ;COMMAND
1533  005613      000                         .BYTE   0               ;PSEL AND A17 AND A16
1534  005614  000000                          .WORD   0               ;WORD COUNT (NEG)
1535  005616  000000                          .WORD   0               ;BUFFER ADDRESS
1536  005620      000                         .BYTE   0               ;SECTOR ADDRESS
1537  005621      000                         .BYTE   0               ;TRACK ADDRESS
1538  005622  000000                          .WORD   0               ;CYLINDER ADDRESS
1539  005624  005540                          .WORD   RM.REG          ;ERROR TABLE POINTER
1540  005626  000000                          .WORD   0               ;STATUS-ERROR INDICATOR
1541                                                                  ;BIT 15 = 1: ERROR OCCURRED
1542                                                                  ;BIT 07 = 1: DONE
1543                                                                  ;BIT 14-10 AND BIT 06-03
1544                                                                  ;INDICATE TYPE OF ERROR
1545
1546
1547  005630      000                 FMTX:   .BYTE   0       ;SPECIAL DPB TO WRITE HEADER OF BAD SPOT
1548  005631      000                         .BYTE   0
1549  005632      000                         .BYTE   0
1550  005633      000                         .BYTE   0
1551  005634  177776                          .WORD   -2      ;ONLY TWO WORDS FOR HEADER
1552  005636  037544                          .WORD   RBUF    ;BUFFER ADDRESS
1553  005640      000                         .BYTE   0       ;SECTOR
1554  005641      000                         .BYTE   0       ;TRACK
1555  005642  000000                          .WORD   0       ;CYLINDER ADDRESS
1556  005644  000000                          .WORD   0       ;ERROR TABLE
1557                                                          ;DONT"SAVE ANYTHING
1558  005646  000000                          .WORD   0       ;STATUS WORD
1559                                  ;PARAMETER POINTER TABLE
1560
1561  005650  005702  001467  001322  TABLE:  PAR1,823.,BEGCYL
1562  005656  005715  000005  001326          PAR2,5.,BEGTRK
```

```
                                        PAR3,823.,ENDCYL
                                        PAR4,5.,ENDTRK,0

                                ;ASCII MESSAGES FOR ADDRESS PARAMETERS

005702  052123  051101  020124  PAR1:   .ASCIZ  @START CYL @
005710  054503  020114  000
005715  123     040524  052122  PAR2:   .ASCIZ  @START TRK @
005722  052040  045522  000040
005730  047105  020104  054503  PAR3:   .ASCIZ  @END CYL @
005736  020114  000
005741  105     042116  052040  PAR4:   .ASCIZ  @END TRK @
005746  045522  000040

                                ;SECTOR BUFFER ADDRESS TABLE

005752  037554                  ADRTBL: .WORD   BUFP             ;ADDRESS OF SECTOR 0
005754  040560                          .WORD   BUFP+<516.*1.>   ;ADDRESS OF SECTOR 1
005756  041564                          .WORD   BUFP+<516.*2.>   ;ADDRESS OF SECTOR 2
005760  042570                          .WORD   BUFP+<516.*3.>   ;ADDRESS OF SECTOR 3
005762  043574                          .WORD   BUFP+<516.*4.>   ;ADDRESS OF SECTOR 4
005764  044600                          .WORD   BUFP+<516.*5.>   ;ADDRESS OF SECTOR 5
005766  045604                          .WORD   BUFP+<516.*6.>   ;ADDRESS OF SECTOR 6
005770  046610                          .WORD   BUFP+<516.*7.>   ;ADDRESS OF SECTOR 7
005772  047614                          .WORD   BUFP+<516.*8.>   ;ADDRESS OF SECTOR 8
005774  050620                          .WORD   BUFP+<516.*9.>   ;ADDRESS OF SECTOR 9
005776  051624                          .WORD   BUFP+<516.*10.>  ;ADDRESS OF SECTOR 10
006000  052630                          .WORD   BUFP+<516.*11.>  ;ADDRESS OF SECTOR 11
006002  053634                          .WORD   BUFP+<516.*12.>  ;ADDRESS OF SECTOR 12
006004  054640                          .WORD   BUFP+<516.*13.>  ;ADDRESS OF SECTOR 13
006006  055644                          .WORD   BUFP+<516.*14.>  ;ADDRESS OF SECTOR 14
006010  056650                          .WORD   BUFP+<516.*15.>  ;ADDRESS OF SECTOR 15
006012  057654                          .WORD   BUFP+<516.*16.>  ;ADDRESS OF SECTOR 16
006014  060660                          .WORD   BUFP+<516.*17.>  ;ADDRESS OF SECTOR 17
006016  061664                          .WORD   BUFP+<516.*18.>  ;ADDRESS OF SECTOR 18
006020  062670                          .WORD   BUFP+<516.*19.>  ;ADDRESS OF SECTOR 19
006022  063674                          .WORD   BUFP+<516.*20.>  ;ADDRESS OF SECTOR 20
006024  064700                          .WORD   BUFP+<516.*21.>  ;ADDRESS OF SECTOR 21
006026  065704                          .WORD   BUFP+<516.*22.>  ;ADDRESS OF SECTOR 22
006030  066710                          .WORD   BUFP+<516.*23.>  ;ADDRESS OF SECTOR 23
006032  067714                          .WORD   BUFP+<516.*24.>  ;ADDRESS OF SECTOR 24
006034  070720                          .WORD   BUFP+<516.*25.>  ;ADDRESS OF SECTOR 25
006036  071724                          .WORD   BUFP+<516.*26.>  ;ADDRESS OF SECTOR 26
006040  072730                          .WORD   BUFP+<516.*27.>  ;ADDRESS OF SECTOR 27
006042  073734                          .WORD   BUFP+<516.*28.>  ;ADDRESS OF SECTOR 28
006044  074740                          .WORD   BUFP+<516.*29.>  ;ADDRESS OF SECTOR 29
006046  075744                          .WORD   BUFP+<516.*30.>  ;ADDRESS OF SECTOR 30
006050  076750                          .WORD   BUFP+<516.*31.>  ;ADDRESS OF SECTOR 31

                        ;REMAINING WORD COUNT TABLE

006052  000402          WCTBL:  .WORD   258.             ;REMAINING WORD COUNT AFTER SECTOR 0
006054  001004                  .WORD   258.+<258.*1.>   ;REMAINING WORD COUNT AFTER SECTOR 1
006056  001406                  .WORD   258.+<258.*2.>   ;REMAINING WORD COUNT AFTER SECTOR 2
006060  002010                  .WORD   258.+<258.*3.>   ;REMAINING WORD COUNT AFTER SECTOR 3
```

```
1619  006062  002412          .WORD   258.+(258.*4.)  ;REMAINING WORD COUNT AFTER SECTOR 4
1620  006064  003014          .WORD   258.+(258.*5.)  ;REMAINING WORD COUNT AFTER SECTOR 5
1621  006066  003416          .WORD   258.+(258.*6.)  ;REMAINING WORD COUNT AFTER SECTOR 6
1622  006070  004020          .WORD   258.+(258.*7.)  ;REMAINING WORD COUNT AFTER SECTOR 7
1623  006072  004422          .WORD   258.+(258.*8.)  ;REMAINING WORD COUNT AFTER SECTOR 8
1624  006074  005024          .WORD   258.+(258.*9.)  ;REMAINING WORD COUNT AFTER SECTOR 9
1625  006076  005426          .WORD   258.+(258.*10.) ;REMAINING WORD COUNT AFTER SECTOR 10
1626  006100  006030          .WORD   258.+(258.*11.) ;REMAINING WORD COUNT AFTER SECTOR 11
1627  006102  006432          .WORD   258.+(258.*12.) ;REMAINING WORD COUNT AFTER SECTOR 12
1628  006104  007034          .WORD   258.+(258.*13.) ;REMAINING WORD COUNT AFTER SECTOR 13
1629  006106  007436          .WORD   258.+(258.*14.) ;REMAINING WORD COUNT AFTER SECTOR 14
1630  006110  010040          .WORD   258.+(258.*15.) ;REMAINING WORD COUNT AFTER SECTOR 15
1631  006112  010442          .WORD   258.+(258.*16.) ;REMAINING WORD COUNT AFTER SECTOR 16
1632  006114  011044          .WORD   258.+(258.*17.) ;REMAINING WORD COUNT AFTER SECTOR 17
1633  006116  011446          .WORD   258.+(258.*18.) ;REMAINING WORD COUNT AFTER SECTOR 18
1634  006120  012050          .WORD   258.+(258.*19.) ;REMAINING WORD COUNT AFTER SECTOR 19
1635  006122  012452          .WORD   258.+(258.*20.) ;REMAINING WORD COUNT AFTER SECTOR 20
1636  006124  013054          .WORD   258.+(258.*21.) ;REMAINING WORD COUNT AFTER SECTOR 21
1637  006126  013456          .WORD   258.+(258.*22.) ;REMAINING WORD COUNT AFTER SECTOR 22
1638  006130  014060          .WORD   258.+(258.*23.) ;REMAINING WORD COUNT AFTER SECTOR 23
1639  006132  014462          .WORD   258.+(258.*24.) ;REMAINING WORD COUNT AFTER SECTOR 24
1640  006134  015064          .WORD   258.+(258.*25.) ;REMAINING WORD COUNT AFTER SECTOR 25
1641  006136  015466          .WORD   258.+(258.*26.) ;REMAINING WORD COUNT AFTER SECTOR 26
1642  006140  016070          .WORD   258.+(258.*27.) ;REMAINING WORD COUNT AFTER SECTOR 27
1643  006142  016472          .WORD   258.+(258.*28.) ;REMAINING WORD COUNT AFTER SECTOR 28
1644  006144  017074          .WORD   258.+(258.*29.) ;REMAINING WORD COUNT AFTER SECTOR 29
1645  006146  017476          .WORD   258.+(258.*30.) ;REMAINING WORD COUNT AFTER SECTOR 30
1646  006150  020100          .WORD   258.+(258.*31.) ;REMAINING WORD COUNT AFTER SECTOR 31
1647
1648                  .EVEN
1649
```

```
                                .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


006152                          $ERRTB:
                                ;ERROR 1

006152  034426                          EM1                     ;RH11 INTERRUPT OCCURRED (RMAS=0)
006154  035545                          DH1
006156  037100                          DT1
006160  037434                          DF1

                                ;ERROR 2

006162  034467                          EM2                     ;UNEXPECTED ATTENTION OCCURRED
006164  035613                          DH2
006166  037116                          DT2
006170  037440                          DF2

                                ;ERROR 3

006172  034525                          EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
006174  035665                          DH3
006176  037132                          DT3
006200  037444                          DF3

                                ;ERROR 4

006202  034563                          EM4                     ;MASSBUS PARITY ERROR (PAR=1)
006204  035736                          DH4
006206  037146                          DT4
006210  037450                          DF4

                                ;ERROR 5

006212  000000                          0                       ;UNUSED
006214  000000                          0
006216  000000                          0
006220  000000                          0

                                ;ERROR 6

006222  034654                          EM6                     ;RH11 DIDN'T RESPOND TO ADDRESSING
006224  036021                          DH6
006226  037164                          DT6
006230  037454                          DF6
```

```
1706
1707                                    ;ERROR 7
1708
1709  006232  000000                            0                       ;UNUSED
1710  006234  000000                            0
1711  006236  000000                            0
1712  006240  000000                            0
1713
1714                                    ;ERROR 10
1715
1716  006242  034716                            EM10                    ;DRIVE OFFLINE
1717  006244  036034                            DH10
1718  006246  037166                            DT10
1719  006250  037460                            DF10
1720
1721                                    ;ERROR 11
1722
1723  006252  034734                            EM11                    ;PERSISTENT DRIVE UNSAFE ERROR
1724  006254  036034                            DH10
1725  006256  037166                            DT10
1726  006260  037460                            DF10
1727
1728                                    ;ERROR 12
1729
1730  006262  034772                            EM12                    ;UNCORRECTABLE MASSBUS PARITY ERROR
1731  006264  036034                            DH10
1732  006266  037166                            DT10
1733  006270  037460                            DF10
1734
1735                                    ;ERROR 13
1736
1737  006272  035035                            EM13                    ;SOFTWARE TIMEOUT
1738  006274  036034                            DH10
1739  006276  037166                            DT10
1740  006300  037460                            DF10
1741
1742                                    ;ERROR 14
1743
1744  006302  035056                            EM14                    ;DRIVE UNSAFE ERROR
1745  006304  036034                            DH10
1746  006306  037166                            DT10
1747  006310  037460                            DF10
1748
1749                                    ;ERROR 15
1750
1751  006312  035101                            EM15                    ;CONTROLLER/DRIVE ERROR DURING WRITE
1752  006314  036034                            DH10
1753  006316  037166                            DT10
1754  006320  037460                            DF10
1755
1756                                    ;ERROR 16
1757
1758  006322  035145                            EM16                    ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1759  006324  036034                            DH10
1760  006326  037166                            DT10
1761  006330  037460                            DF10
```

```
1762
1763                                    ;ERROR 17
1764
1765  006332  035217                            EM17            ;RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE
1766  006334  036263                            DH17
1767  006336  037240                            DT17
1768  006340  037474                            DF17
1769
1770                                    ;ERROR 20
1771
1772  006342  035275                            EM20            ;DATA ERROR DURING WRITE CHECK
1773  006344  036332                            DH20
1774  006346  037252                            DT20
1775  006350  037500                            DF20
1776
1777                                    ;ERROR 21
1778
1779  006352  035333                            EM21            ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
1780  006354  036034                            DH10
1781  006356  037166                            DT10
1782  006360  037460                            DF10
1783
1784                                    ;ERROR 22
1785
1786  006362  035404                            EM22            ;'HCE' ERROR VERIFYING HEADERS
1787  006364  036034                            DH10
1788  006366  037166                            DT10
1789  006370  037460                            DF10
1790
1791                                    ;ERROR 23
1792
1793  006372  035453                            EM23            ;CYLINDER FIELD IN HEADER IS NOT CORRECT
1794  006374  036627                            DH23
1795  006376  037334                            DT23
1796  006400  037520                            DF23
1797
1798                                    ;ERROR 24
1799
1800  006402  035523                            EM24            ;WRITE CHECK ERROR
1801  006404  036725                            DH24
1802  006406  037346                            DT24
1803  006410  037524                            DF24
1804
1805                                    ;;*****************************************************************
1806
1807                                    .SBTTL  MAIN PROGRAM
1808
1809                                    ;;*****************************************************************
1810
1811  006412  012737  177777  001400    BEGIN:  MOV     #-1,CHGADR      ;SET CHANGE 'RH70 BUS ADDRESS' INDICATOR
1812  006420  000406                            BR      BEGIN2          ;START THE PROGRAM
1813  006422  005037  001400            BEGIN1: CLR     CHGADR          ;CLEAR THE 'CHANGE RH70 ADDRESS' INDICATOR
1814                                    ;;*****************************************************************
1815  006426  000240                    TST1:   NOP
1816  006430  012737  000001  001212            MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1817  006436  000005                    BEGIN2: RESET                   ;CLEAR THE BUS
```

```
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
006440  012706  001114          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
006444  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
006446  022706  001154          CMP     #SWR,R6 ;;DONE?
006452  001374                  BNE     .-6             ;;LOOP BACK IF NO
006454  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
006460  012737  020066  000030  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
006466  012737  000340  000032  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
006474  012737  023762  000034  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
006502  012737  000340  000036  MOV     #340,@#TRAPVEC+2;LEVEL 7
006510  012737  023224  000024  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
006516  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
006524  005037  001174          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
006530  112737  000001  001131  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
006536  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
006542  012737  006576  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
006550  012737  177570  001154  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
006556  012737  177570  001156  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
006564  022777  177777  172362  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
006572  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
006574  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
006576  012716  006604  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
006602  000002                  RTI
006604  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
006612  012737  000174  001156          MOV     #DISPREG,DISPLAY
006620  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

006624  005037  001214          CLR     $PASS           ;;CLEAR PASS COUNT
006630  132737  000200  001227  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
006636  001403                  BEQ     67$             ;;YES,USE NON-APT SWITCH
006640  012737  001230  001154  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
006646                  67$:
006646  000005                  RESET                   ;CLEAR WORLD
006650  012737  000240  000036  MOV     #PR5,@#TRAPVEC+2  ;CHANGE TRAP PRIORITY BACK TO 5
006656  012737  000240  000032  MOV     #PR5,@#EMTVEC+2 ;CHANGE EMT PRIORITY BACK TO 5
006664  012737  006422  001376  MOV     #BEGIN1,CNTLC   ;'CONTROL C' ADDRESS
006672  005227  177777          INC     #-1             ;FIRST START ?
006676  001002                  BNE     1$              ;BR IF NOT
006700  104401  037554          TYPE    ,TITLE          ;ADRS OF 'TITLE' MESSAGE
                                                        ;4 LINES HAS DELETED 3/14/77
006704  004737  021772  1$:     JSR     PC,$TKINT       ;TURN ON THE KEYBOARD INTERRUPT
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
006710  005737  000042          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
006714  001012                  BNE     68$             ;;BRANCH IF YES
006716  123727  001226  000001  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
006724  001406                  BEQ     68$             ;;BRANCH IF YES
006726  023727  001154  000176  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
006734  001005                  BNE     69$             ;;BRANCH IF NO
006736  104406                  GTSWR                   ;;GET SOFT-SWR SETTINGS
006740  000403                  BR      69$
006742  112737  000001  001150  68$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
```

```
1874  006750                          69$:
1875  006750  005227  177777                  INC     #-1             ;CHECK FOR FIRST START
1876  006754  001035                          BNE     SETVEC          ;BR IF NOT
1877  006756  004737  040076                  JSR     PC,BUSADR       ;CHECK THE RH11 ADDRESS
1878  006762  013737  001272  024214          MOV     $RMADR,RMADR    ;RH11 ADDRESS
1879  006770  013737  001274  024216          MOV     $RMVEC,RMVEC    ;RH11 VECTOR ADDRESS
1880  006776  105737  001226                  TSTB    $ENV            ;RUN UNDER APT MODE?
1881  007002  001422                          BEQ     SETVEC          ;NO,DONOT LOAD FROM E-TABLE
1882  007004  105737  001256                  TSTB    $VECT1          ;UUT INTERRUPT VECTOR
1883  007010  001403                          BEQ     .+10            ;NOT CHANGE IF = 0
1884  007012  113737  001256  001274          MOVB    $VECT1,$RMVEC   ;NEWVALUE
1885  007020  005737  001262                  TST     $BASE           ;UUT BASE REGISTER ADDRESS
1886  007024  001403                          BEQ     .+10            ;NOT CHANGE IF = 0
1887  007026  013737  001262  001272          MOV     $BASE,$RMADR    ;NEW BASE ADDRESS
1888  007034  013737  001272  024214          MOV     $RMADR,RMADR    ;RH11/70 ADDRESS
1889  007042  013737  001274  024216          MOV     $RMVEC,RMVEC    ;RH11/70 VECTOR
1890
1891                                  ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
1892                                  ;       PROGRAM WILL USE
1893
1894  007050  004737  016104          SETVEC: JSR     PC,ST.CLK       ;START THE CLOCK
1895  007054  004737  024232                  JSR     PC,RMINIT       ;INITIALIZE THE RM03 DRIVER
1896  007060  012737  177777  024154          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
1897  007066  005227  177777                  INC     #-1             ;SEE IF FIRST START
1898  007072  001404                          BEQ     11$             ;BR IF YES
1899  007074  032777  000004  172052          BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
1900  007102  001101                          BNE     10$             ;BR IF NOT
1901  007104  012737  000340  177776  11$:    MOV     #PR7,PS         ;SET PRIORITY TO 7
1902  007112  005004                          CLR     R4              ;DRIVE TABLE POINTER
1903  007114  104401  001203                  TYPE    ,$CRLF          ;CR-LF
1904  007120  104401  032436                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
1905  007124                          1$:
1906  007124  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
1907                                          ;;TYPE DRIVE NUMBER
1908  007126  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1909  007130     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1910  007131     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1911  007132  104401  032501                  TYPE    ,LIN4SP         ;SPACES
1912  007136  105764  024066                  TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
1913  007142  100416                          BMI     4$              ;BR IF UNSAFE
1914  007144  001020                          BNE     5$              ;BR IF ONLINE
1915  007146  105764  024076                  TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
1916  007152  001404                          BEQ     2$              ;BR IF NONEXISTENT
1917  007154  100006                          BPL     3$              ;BR IF OFFLINE
1918  007156  104401  032355                  TYPE    ,NOTRM          ;DRIVE NOT AN RM03
1919  007162  000443                          BR      9$              ;CHECK NEXT DRIVE
1920  007164  104401  032372          2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
1921  007170  000440                          BR      9$              ;CHECK NEXT DRIVE
1922  007172  104401  032334          3$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
1923  007176  000410                          BR      6$              ;PRINT DRIVE TYPE
1924  007200  104401  032407          4$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
1925  007204  000405                          BR      6$              ;PRINT DRIVE TYPE
1926  007206  104401  032345          5$:     TYPE    ,UNTON          ;DRIVE ONLINE
1927  007212  156437  024202  001404          BISB    ATABIT(R4),DEVMP        ;SET DEVICE MAP
1928  007220  104401  032503          6$:     TYPE    ,LINSP          ;SPACES
1929  007224  012737  032417  007270          MOV     #RM03B,8$       ;ADDRESS OF RP04 MESSAGE
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
1930  007232  132764  000001  024076          BITB    #BIT00,DRVTYP(R4)  ;RM03 ?
1931  007240  001012                          BNE     7$                 ;BR IF YES
1932  007242  012737  032424  007270          MOV     #RP05,8$           ;ADDRESS OF RP05 MESSAGE
1933  007250  132764  000002  024076          BITB    #BIT01,DRVTYP(R4)  ;RP05 ?
1934  007256  001003                          BNE     7$                 ;BR IF YES
1935  007260  012737  032431  007270          MOV     #RM03A,8$          ;ADDRESS OF RM03 MESSAGE
1936  007266  104401                  7$:     TYPE                       ;TYPE THE DRIVE TYPE MESSAGE
1937  007270  000000                  8$:     .WORD   0                  ;MESSAGE ADDRESS HERE.
1938  007272  104401  001203          9$:     TYPE    ,$CRLF             ;CR-LF
1939  007276  005204                          INC     R4                 ;INCREMENT DRIVE NUMBER/TABLE POINTER
1940  007300  020427  000010                  CMP     R4,#8.             ;FINISHED ?
1941  007304  001307                          BNE     1$                 ;BR IF NOT
1942  007306  104401  001203          10$:    TYPE    ,$CRLF             ;CR-LF
1943                                  ;
                                    ;CODING FOR APT---LOAD PARAMETERS FROM E-TABLE
1944  007312  105737  001150          APT:    TSTB    $AUTOB             ;RUN UNDER APT ?
1945  007316  001544                          BEQ     M1                 ;NO,DON'T BOTHER
1946  007320  005037  001410                  CLR     EDIT    ;CLEAR THE TEXT LAST TRACK FLAG
1947  007324  005737  001264                  TST     $DEVM   ;DEVICE MAP AVAILABLE FROM APT MAIL BOX
1948  007330  001406                          BEQ     1$                 ;BRANCH IF NONE
1949  007332  013737  001264  001404          MOV     $DEVM,DEVMP        ;LOAD IT FORM MAIL BOX
1950  007340  042737  177400  001404          BIC     #177400,DEVMP      ;ALLOW MAX 8 DRIVES
1951  007346  005737  001404          1$:     TST     DEVMP              ;CHECK DEVICE MAP
1952  007352  001002                          BNE     M5FX               ;BRANCH IF DRIVES ASSIGNED
1953  007354  000137  014506                  JMP     $EOP               ;OTHERWISE,EXIT
1954  007360  005037  001326          M5FX:   CLR     BEGTRK             ;ALWAY STARTS FROM 0 TRK
1955  007364  005037  001322                  CLR     BEGCYL             ;ALWAYS STARTS FROM 0 CYLINDER
1956  007370  012737  001466  001320          MOV     #822.,ENDCYL       ;LAST CYLINDER
1957  007376  012737  000004  001324          MOV     #4,ENDTRK          ;LAST TRACK
1958  007404  005037  001402                  CLR     WRAP               ;CLEAR WRAP AROUND FLAG
1959  007410  012737  177777  001316          MOV     #-1,MODE           ;SET ORMAT AND VERIFY MODE
1960  007416  005737  001266                  TST     $CDW1              ;APT PARAMETER = 0
1961  007422  001402                          BEQ     .+6                ;YES,DEFAULT TO -1
1962  007424  005037  001316                  CLR     MODE               ;NO,SET TO CHECK MODE
1963  007430  012737  177777  001362          MOV     #-1,SEC30          ;SET 32 SECTOR
1964  007436  005737  001270                  TST     $CDW2              ;APT PARAMETER =0
1965  007442  001012                          BNE     1$                 ;NO,30 SECTOR
1966  007444  012737  020100  001354          MOV     #<258.*32.>,WC     ;COUNT OF WORDS
1967  007452  012737  157700  001356          MOV     #-<258.*32.>,MWC           ;WORD COUNT
1968  007460  012737  000037  001364          MOV     #31.,MAXSEC        ;MAX 32 SECTOR
1969  007466  000413                          BR      2$
1970  007470  012737  017074  001354  1$:     MOV     #<258.*30.>,WC     ;COUNT OF WORDS
1971  007476  012737  160704  001356          MOV     #-<258.*30.>,MWC           ;WORD COUNT
1972  007504  005037  001362                  CLR     SEC30              ;30 SECTOR
1973  007510  012737  000035  001364          MOV     #29.,MAXSEC        ;MAX 30 SECTOR
1974  007516  005737  001316          2$:     TST     MODE               ;CHECK MODE ?
1975  007522  001006                          BNE     .+16               ;NO
1976  007524  012737  133331  001340          MOV     #133331,PATA       ;INITIATE PATTERN
1977  007532  012737  133331  001342          MOV     #133331,PATB       ;INITIATE PATTERN
1978  007540  012737  000002  001336          MOV     #2,PATSEL          ;WORSE CASE
1979  007546  012700  000001                  MOV     #1,R0              ;UUT STARTS AT DRIVE 0
1980  007552  005001                          CLR     R1                 ;DRIVE NUMBER
1981  007554  030037  001404          3$:     BIT     R0,DEVMP           ;BIT OF DEVICE MAP SET ?
1982  007560  001005                          BNE     4$      ;FIND THE DRIVE # FROM DEVICE MAP
1983  007562  000241                          CLC
1984  007564  006100                          ROL     R0      ;NEXT DRIVE
1985  007566  003416                          BLE     6$      ;BRANCH IF ALL DRIVES ARE DONE
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
1986  007570  005201                        INC     R1              ;INCREMENT DRIVE #
1987  007572  000770                        BR      3$      ;LOOP
1988  007574  010137  001220        4$:     MOV     R1,DRIVE        ;LOAD R1 INTO DRIVE
1989  007600  040037  001404                BIC     R0,DEVMP        ;CLEAR THE BIT FROM DEVICE MAP
1990  007604  105761  024066                TSTB    DRVSTA(R1)      ;DRIVE ON LINE ?
1991  007610  003003                        BGT     5$      ;BRANCH IF DRIVE ON LINE
1992  007612  005237  001216                INC     $DEVCT  ;INCREMENT DEVICE COUNT TO CHANGE MAIL BOX
1993  007616  000756                        BR      3$      ;LOOP BACK
1994  007620  000137  010542        5$:     JMP     M5      ;START TO FORMAT OR CHECK
1995  007624  000137  014506        6$:     JMP     $EOP    ;END OF PASS FOR ALL DRIVES ON $DEVM
1996
1997
1998                                ;SEE WHICH MODE THE PROGRAM IS TO BE RUN IN.
1999                                ;MODES ARE: 'FORMAT & VERIFY' OR 'CHECK FORMAT'
2000
2001  007630  005037  001316        M1:     CLR     MODE            ;SET MODE TO 'CHECK FORMAT'
2002  007634  104401  032713                TYPE    ,MMODE          ;TYPE 'PROGRAM MODE'
2003  007640  104411                        RDLIN                   ;READ THE KEYBOARD
2004  007642  012601                        MOV     (SP)+,R1        ;GET ADDRESS OF INPUT
2005  007644  105711                        TSTB    (R1)            ;'CR' ENTERED (DEFAULT)
2006  007646  001414                        BEQ     2$              ;BR IF YES
2007  007650  122711  000103                CMPB    #'C,(R1)        ;'CHECK' ?
2008  007654  001406                        BEQ     1$              ;BR IF YES
2009  007656  122711  000106                CMPB    #'F,(R1)        ;'FORMAT' ?
2010  007662  001411                        BEQ     3$              ;BR IF YES
2011  007664  104401  001202                TYPE    ,$QUES          ;NO CORRECT ENTRY
2012  007670  000757                        BR      M1              ;TRY AGAIN
2013  007672  104401  032766        1$:     TYPE    ,MHECK          ;TYPE REST OF 'CHECK'
2014  007676  000410                        BR      M1A             ;GET STARTING ADDRESS
2015  007700  104401  032745        2$:     TYPE    ,MFORMT         ;TYPE DEFAULT MESSAGE
2016  007704  000402                        BR      4$              ;SET UP MODE
2017  007706  104401  033001        3$:     TYPE    ,MORMAT         ;TYPE REST OF 'FORMAT'
2018  007712  012737  177777  001316  4$:     MOV     #-1,MODE        ;SET MODE TO 'FORMAT & VERIFY'
2019
2020                                ;FIND OUT IF FORMAT IS TO BE IN 30 OR 32 SECTOR MODE
2021
2022  007720  104401  033021        M1A:    TYPE    ,MSIZE          ;TYPE FORMAT MODE REQUEST
2023  007724  104411                        RDLIN                   ;READ THE KEYBOARD
2024  007726  012601                        MOV     (SP)+,R1        ;ADDRESS OF ENTRY
2025  007730  122711  000131                CMPB    #'Y,(R1)        ;IS ENTRY A 'Y' ?
2026  007734  001410                        BEQ     1$              ;BR IF IT IS
2027  007736  122711  000116                CMPB    #'N,(R1)        ;IS ENTRY A 'N' ?
2028  007742  001424                        BEQ     2$              ;BR IF IT IS
2029  007744  105711                        TSTB    (R1)            ;DEFAULT MODE ?
2030  007746  001403                        BEQ     1$              ;BR IF IT IS
2031  007750  104401  001202                TYPE    ,$QUES          ;TYPE '?'
2032  007754  000761                        BR      M1A             ;TRY AGAIN
2033  007756  104401  033072        1$:     TYPE    ,MSEC22         ;TYPEOUT MODE SELECTED
2034  007762  012737  020100  001354        MOV     #<258.*32.>,WC  ;TRACK SIZE IN 32 SECTOR MODE
2035  007770  012737  157700  001356        MOV     #-<258.*32.>,MWC  ;2'S COMPLEMENT WORD COUNT
2036  007776  012737  177777  001362        MOV     #-1,SEC30       ;32 SECTOR INDICATOR
2037  010004  012737  000037  001364        MOV     #31.,MAXSEC     ;MAX SECTOR ADDRESS IN 32 SECTOR MODE
2038  010012  000415                        BR      M1B             ;CONTINUE
2039  010014  104401  033151        2$:     TYPE    ,MSEC20         ;TYPE OUT 30 SECTOR MODE SELECTED
2040  010020  012737  017074  001354        MOV     #<258.*30.>,WC  ;TRACK SIZE IN 30 SECTOR MODE
2041  010026  012737  160704  001356        MOV     #-<258.*30.>,MWC  ;2'S COMPLEMENT WORD COUNT
```

```
2042  010034  005037  001362                  CLR     SEC30           ;30 SECTOR INDICATOR
2043  010040  012737  000035  001364          MOV     #29.,MAXSEC     ;MAX SECTOR ADDRESS IN 30 SECTOR MODE
2044  010046  005037  001326          M1B:    CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
2045  010052  005037  001322                  CLR     BEGCYL          ;CLEAR BEGINNING CYLINDER ADDRESS
2046  010056  005037  001402                  CLR     WRAP            ;CLEAR WRAP AROUND FLAG
2047  010062  012737  000004  001324          MOV     #4.,ENDTRK      ;SETUP END TRACK ADDRESS
2048  010070  012737  000002  001336          MOV     #2,PATSEL       ;SETUP FOR WORST CASE PATTERN
2049  010076  112737  177777  005610          MOVB    #-1,FMTDPB      ;SETUP INVALID DRIVE NUMBER
2050  010104  000400                          BR      M0              ;BRANCH TO START
2051
2052                          ;GO FIND OUT WHAT DRIVE
2053
2054  010106  012737  016324  001376  M0:     MOV     #0ENTER,CNTLC   ;CONTROL C ABORT ENTRANCE
2055  010114  005037  001126                  CLR     $ERTTL          ;CLEAR THE ERROR ACCUMULATOR
2056  010120  004737  021772                  JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
2057  010124  104401  032455                  TYPE    ,MUNIT          ;ASK FOR DRIVE NUMBER
2058  010130  104411                          RDLIN                   ;READ THE KEYBOARD
2059  010132  012601                          MOV     (SP)+,R1        ;ADDRESS OF ENTRY
2060  010134  004537  017656                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
2061  010140  010164                          3$                      ;ILLEGAL CHARACTER
2062  010142  010154                          2$                      ;CARRIAGE RETURN
2063  010144  010164                          3$                      ;" "
2064  010146  010154                          2$                      ;"-"
2065  010150  010172                          4$                      ;DİGIT 0-7
2066  010152  010164                          3$                      ;DIGIT 8-9
2067  010154  005002                  2$:     CLR     R2              ;SELECT DRIVE ZERO
2068  010156  104401  032477                  TYPE    ,MDRVD          ;GO TYPE DEFAULT DRIVE NUMBER
2069  010162  000403                          BR      4$              ;GO SEE IF DRIVE ZERO IS THERE
2070  010164  104401  001202          3$:     TYPE    ,$QUES          ;TYPE '?'
2071  010170  000746                          BR      M0              ;ASK FOR DRIVE NUMBER AGAIN
2072  010172  010237  001220          4$:     MOV     R2,DRIVE        ;SAVE DRIVE NUMBER
2073  010176  005702                          TST     R2              ;SEE IF DRIVE 0
2074  010200  001004                          BNE     5$              ;BR IF NOT
2075  010202  122737  000011  000041          CMPB    #11,41          ;PROGRAM LOADED FROM AN RM03 ?
2076  010210  001431                          BEQ     7$              ;BR IF IT WAS, CAN'T FORMAT THE DRIVE
2077  010212  004737  016104          5$:     JSR     PC,ST.CLK       ;START THE CLOCK
2078  010216  004737  024232                  JSR     PC,RMINIT       ;GO SEE WHAT DRIVES ARE AVAILABLE
2079  010222  012737  177777  024154          MOV     #-1,SAVEFG      ;SAVE THE REGISTERS
2080  010230  012737  177777  024156          MOV     #-1,SEEKFG      ;SET 'NO OPTIMIZATION' FLAG
2081  010236  005037  177776                  CLR     PS              ;SET PRIORITY BACK TO ZERO
2082  010242  105762  024066                  TSTB    DRVSTA(R2)      ;LOOK AT DRIVE STATUS
2083  010246  003015                          BGT     8$              ;BRANCH IF ONLINE
2084  010250  001403                          BEQ     6$              ;BR IF DRIVE NOT AVAILABLE
2085  010252  104401  032661                  TYPE    ,MUSDR          ;'DRIVE UNSAFE'
2086  010256  000713                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
2087  010260  105762  024076          6$:     TSTB    DRVTYP(R2)      ;A DRIVE PRESENT?
2088  010264  001003                          BNE     7$              ;BR IF SO
2089  010266  104401  032560                  TYPE    ,MDRNP          ;TYPE 'DRIVE NOT PRESENT'
2090  010272  000705                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
2091  010274  104401  032605          7$:     TYPE    ,MER11          ;'DRIVE NOT AVAILABLE'
2092  010300  000702                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
2093  010302  123737  001220  005610  8$:     CMPB    DRIVE,FMTDPB    ;SAME DRIVE AS LAST TIME ?
2094  010310  001422                          BEQ     M2              ;BR IF IT IS
2095  010312  113737  001220  005610          MOVB    DRIVE,FMTDPB    ;SETUP DRIVE ADDRESS
2096  010320  012737  000632  001320          MOV     #410.,ENDCYL    ;SETUP FOR RM03
2097  010326  132762  000003  024076          BITB    #BIT00!BIT01,DRVTYP(R2)  ;SEE IF DRIVE RM03
```

```
2098  010334  001003                        BNE     9$              ;BR IF EITHER
2099  010336  012737  001466  001320        MOV     #822.,ENDCYL    ;SETUP ENDING CYLINDER FOR RM03
2100  010344  005037  001326          9$:   CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
2101  010350  005037  001322                CLR     BEGCYL          ;CLEAR STARTING CYLINDER ADDRESS
2102  010354  000450                        BR      M2              ;GET ADDRESS LIMITS FROM THE OPERATOR
2103
2104                                  ;GET ADDRESS LIMITS
2105
2106  010356  104401  032506          M2:   TYPE    ,ENTADR         ;'ENTER ADDRESS LIMITS'
2107  010362  004737  016502                JSR     PC,PARENT       ;GET THE ADDRESS LIMITS
2108  010366  005037  001402                CLR     WRAP            ;CLEAR WRAP FLAG
2109  010372  023737  001320  001322        CMP     ENDCYL,BEGCYL   ;SEE IF ENDING CYLINDER EQ TO GT THAN BEGINNING
2110  010400  001402                        BEQ     2$              ;BR IF ON THE SAME CYLINDER
2111  010402  103410                        BLO     4$              ;BR IF LESS
2112  010404  000413                        BR      M4              ;TO CHECK DRIVE
2113  010406  023737  001324  001326  2$:   CMP     ENDTRK,BEGTRK   ;SEE IF ENDING TRACK EQ OR GT THAN BEGINNING
2114  010414  103007                        BHIS    M4              ;BR IF YES
2115  010416  104401  033250          3$:   TYPE    ,MADRER         ;INVALID ADDRESS ENTERED
2116  010422  000755                        BR      M2              ;TRY AGAIN
2117  010424  012737  177777  001402  4$:   MOV     #-1,WRAP        ;SET WRAP AROUND FLAG
2118  010432  000400                        BR      M4
2119
2120                                  ;GO GET DATA PATTERN FOR FORMAT
2121
2122  010434  005737  001316          M4:   TST     MODE            ;VERIFY MODE ?
2123  010440  001007                        BNE     M4A             ;NO
2124  010442  012737  133331  001340        MOV     #133331,PATA    ;VERIFY MODE USE WOREST CASE DATA
2125  010450  012737  133331  001342        MOV     #133331,PATB    ;INITIAL THE DATA PATTERN -SHIFT 3 BITS
2126  010456  000431                        BR      M5              ;START TO FORMAT AND CHECK
2127  010460  104401  033352          M4A:  TYPE    ,MSELP          ;GO TYPE 'SELECT PATTERN'
2128  010464  104411                        RDLIN                   ;READ THE KEYBOARD
2129  010466  012601                        MOV     (SP)+,R1        ;ENTRY ADDRESS
2130  010470  004537  017656                JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
2131  010474  010530                        3$                      ;ILLEGAL CHARACTER
2132  010476  010510                        1$                      ;CARRIAGE RETURN
2133  010500  010530                        3$                      ;" "
2134  010502  010510                        1$                      ;"."
2135  010504  010522                        2$                      ;DIGIT 0-7
2136  010506  010530                        3$                      ;DIGIT 8-9
2137  010510  104401  033502          1$:   TYPE    ,MPATD          ;TYPE DEFAULT PATTERN
2138  010514  012702  000002                MOV     #2,R2           ;WORST CASE PATTERN
2139  010520  000406                        BR      4$              ;GO SAVE PATTERN
2140  010522  020227  000002          2$:   CMP     R2,#2           ;IS # LARGER THAN 2
2141  010526  101403                        BLOS    4$              ;BRANCH IF NOT
2142  010530  104401  001202          3$:   TYPE    ,$QUES          ;TYPE '?'
2143  010534  000737                        BR      M4              ;RETYPE LINE
2144  010536  010237  001336          4$:   MOV     R2,PATSEL       ;SAVE PATTERN SELECTED
2145
2146                                  ;GO TYPE 'STARTING FORMAT ON DRIVE N'
2147
2148  010542  012737  016344  000060  M5:   MOV     #NEWSVC,@#TKVEC ;VECTOR FOR CONTROL-O
2149  010550  005037  177776                CLR     @#PSW           ;LEVEL 0
2150  010554  005737  001316                TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
2151  010560  001403                        BEQ     1$              ;BR IF 'CHECK' MODE
2152  010562  104401  033515                TYPE    ,MSFOU          ;TYPE 'STARTING FORMAT ON DRIVE N'
2153  010566  000402                        BR      2$
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
2154  010570  104401  033552          1$:     TYPE    ,MSCHK          ;TYPE 'STARTING CHECK ON DRIVE N'
2155  010574                          2$:
2156  010574  013746  001220                  MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
2157                                                                  ;;TYPE DRIVE NUMBER
2158  010600  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2159  010602     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
2160  010603     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2161  010604  104401  001203                  TYPE    ,$CRLF          ;CR-LF
2162
2163                                  ;THIS CODE RETRIEVES THE BAD SPOT FILE FROM THEPACK (IF FORMATTED PACK)
2164                                  ;SET UP DPB BLOCK READ CYL 822. TRK 4
2165
2166  010610  005037  001346          RETBAD: CLR     RETRY           ;CLEAR THE RETRY FLAG
2167  010614  013737  001220  005610          MOV     DRIVE,FMTDPB    ;LOAD DRIVE #
2168  010622  012737  157700  005614          MOV     #-(32.*258.),FMTDPB+4 ;WORD COUNT FULL TRACK INCLUDING HEAD
2169  010630  012737  037554  005616          MOV     #BUFP,FMTDPB+6  ;BUFFER ADDRESS
2170  010636  105037  005620                  CLRB    FMTDPB+10       ;SECTOR 0
2171  010642  112737  000004  005621          MOVB    #4,FMTDPB+11    ;TRACK 4
2172  010650  012737  001466  005622          MOV     #822.,FMTDPB+12 ;CYL 822
2173  010656  112737  000173  005612          MOVB    #RDHD,FMTDPB+2  ;READ HEAD AND DATA COMMAND
2174  010664  012737  177777  001406          MOV     #-1,PACK        ;RESET THE MFG AVAILABLE FLAG
2175  010672  112737  000143  005612          MOVB    #SETFMT,FMTDPB+2  ;SET 16 BIT MODE
2176  010700  112737  000020  005611          MOVB    #20,FMTDPB+1    ;FMT SET
2177  010706  004037  024736          1$:     JSR     R0,RM03         ;SET THE FORMAT BIT
2178  010712  005610                          FMTDPB
2179  010714  000774                          BR      1$
2180  010716  005737  005626          2$:     TST     FMTDPB+16       ;THE COMMAND IS DONE ?
2181  010722  001775                          BEQ     2$              ;BRANCH IF NOT
2182  010724  112737  000173  005612          MOVB    #RDHD,FMTDPB+2  ;RELOAD THE READ HEAD AND DATA COMMAND
2183  010732  012737  010732  001124          MOV     #.,$LPERR       ;LOOP ON ERROR ADDRESS
2184  010740  012706  001100                  MOV     #STACK,SP       ;RESET STACK POINT
2185  010744  004037  024736          3$:     JSR     R0,RM03         ;READ THE LAST TRACK
2186  010750  005610                          FMTDPB
2187  010752  000774                          BR      3$              ;BRANCH IF QUEUE FAIL
2188  010754  005737  005626          4$:     TST     FMTDPB+16       ;ALL DONE
2189  010760  001775                          BEQ     4$              ;BRANCH IF NOT
2190  010762  100023                          BPL     6$              ;BRANCH IF NO ERROR
2191  010764  012737  011010  001174          MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
2192  010772  004737  015230                  JSR     PC,ERINDX       ;SEE WHICH ERROR
2193  010776  104010                          ERROR   10
2194  011000  104011                          ERROR   11
2195  011002  104012                          ERROR   12
2196  011004  104013                          ERROR   13
2197  011006  104014                          ERROR   14
2198  011010  004737  015330          5$:     JSR     PC,LOP.CK       ;CHECK LOOP ON ERROR
2199  011014  022737  000003  001346          CMP     #3,RETRY        ;DONT' TRY OVER 3 TIMES
2200  011022  103462                          BLO     8$              ;
2201  011024  005237  001346                  INC     RETRY
2202  011030  000745                          BR      3$              ;TRY AGAIN
2203  011032  005037  001174          6$:     CLR     $ESCAPE         ;
2204  011036  005037  001346                  CLR     RETRY
2205  011042  005737  005626                  TST     FMTDPB+16       ;ANY ERROR ?
2206  011046  100450                          BMI     8$              ;BRANCH IF ANY
2207  011050  032737  140000  037554          BIT     #BIT15!BIT14,@#BUFP ;HEADER INFORMATION  OK ?
2208  011056  001444                          BEQ     8$              ;BRANCH IF NOT
2209  011060  032737  010000  037554          BIT     #BIT12,@#BUFP   ;FMT BET MUST BE SET
```

```
2210  011066  001440                        BEQ     8$                ;BRANCH IF NOT SET
2211  011070  042737  150000  037554        BIC     #150000,@#BUFP    ;GET THE CYLINDER ADDRESS
2212  011076  022737  001466  037554        CMP     #822.,@#BUFP      ;ON THE LAST CYLINDER ?
2213  011104  001031                        BNE     8$                ;BRANCH IF NOT
2214  011106  022737  002000  037556        CMP     #2000,@#BUFP+2    ;ON TRACK 4 AND SECTOR 0
2215  011114  001025                        BNE     8$                ;BRANCH IF NOT
2216  011116  005737  037560                TST     @#BUFP+4          ;SERIAL NUMBER SHOULD NOT 0
2217  011122  001003                        BNE     7$                ;BRANCH IF NOT 0
2218  011124  005737  037562                TST     @#BUFP+6          ;SECOND WORD OF SERIAL NUMBER
2219  011130  001417                        BEQ     8$                ;BRANCH IF ZERO
2220  011132  005737  037564        7$:     TST     @#BUFP+10         ;THE THIRD WORD MUST BE 0
2221  011136  001014                        BNE     8$                ;BRANCH IF NOT
2222  011140  022737  177777  037566        CMP     #-1,@#BUFP+12     ;AN ALIGNMENT PACK ?
2223  011146  001002                        BNE     .+6               ;BRANCH IF NOT
2224  011150  000137  015212                JMP     REJEC1
2225  011154  005737  037566                TST     @#BUFP+12         ;A DATA PACK ?
2226  011160  001003                        BNE     8$                ;BRANCH IF NOT IDITENTIFIED
2227  011162  012737  000400  001406        MOV     #400,PACK         ;SET MFG BAD SPOT FILE AVAILABLE FLAG
2228  011170  000240                8$:     NOP                       ;DONE
2229                                  ;THIS CODE INITIALIZES THE BAD16,BAD18,USTAB,USSAV TABLES
2230                                  ;
2231
2232  011172  012706  001100        TABINT: MOV     #STACK,SP         ;INITIAL STACK POINT
2233  011176  005046                        CLR     -(SP)             ;TERMINATOR
2234  011200  012746  004430                MOV     #USSAV,-(SP)      ;USSAV TABLE ADDRESS
2235  011204  012746  002420                MOV     #BAD18,-(SP)      ;MFG 18 BIT FILE ADDRESS
2236  011210  012746  001414                MOV     #BAD16,-(SP)      ;MFG 16 BIT FILE ADDRESS
2237  011214  012703  003424                MOV     #USTAB,R3         ;USER BAD SPOT FILE ADDRESS
2238  011220  012704  000004        1$:     MOV     #4,R4             ;FIRST 4 WORDS SET TO 0 TEMPERARY
2239  011224  005023                2$:     CLR     (R3)+
2240  011226  005304                        DEC     R4
2241  011230  001375                        BNE     2$                ;BRANCH IF NOT DONE
2242  011232  012704  000374                MOV     #252.,R4          ;SET -1 TO ALL THE REST LOCATIONS
2243  011236  012705  177777                MOV     #-1,R5
2244  011242  010523                3$:     MOV     R5,(R3)+
2245  011244  005304                        DEC     R4                ;DECREMENT WORD COUNT
2246  011246  001375                        BNE     3$                ;BRANCH IF NOT DONE
2247  011250  012603                        MOV     (SP)+,R3          ;NEXT TABLE
2248  011252  001362                        BNE     1$                ;LOOPING BACK IF NOT TERMINATOR
2249
2250
2251                                  ;THIS CODE LOADS TABLE BAD16,BAD18,USSAV,FROM PACK (IF FORMATTED PACK)
2252                                  ;
2253  011254  005737  001406        TABLD:  TST     PACK              ;BAD SPOT FILE AVAILABLE FROM PACK ?
2254  011260  003440                        BLE     5$                ;BRANCH IF NOT
2255  011262  012702  037560                MOV     #BUFP+4,R2        ;LOAD BAD16 TABLE FIRST
2256  011266  012703  001414                MOV     #BAD16,R3         ;TABLE ADDRESS
2257  011272  012704  000400                MOV     #256.,R4          ;WORD COUNT
2258  011276  012223                1$:     MOV     (R2)+,(R3)+       ;LOAD THE WHOLE SECTOR
2259  011300  005304                        DEC     R4                ;BRANCH IF NOT DONE
2260  011302  001375                        BNE     1$
2261  011304  012702  040564                MOV     #BUFP+520.,R2     ;258 X 2 + 2 X 2 + BUFP
2262  011310  012703  002420                MOV     #BAD18,R3         ;TABLE ADDRESS
2263  011314  012704  000400                MOV     #256.,R4          ;WORD COUNT
2264  011320  012223                2$:     MOV     (R2)+,(R3)+       ;LOAD INTO TABLE
2265  011322  005304                        DEC     R4                ;DECREMENT WORD COUNT
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
011324  001375                  BNE     2$              ;BRANCH IF NOT DONE
011326  012702  052634          MOV     #BUFP+4+<11.*516.>,R2 ;SECTOR 11
011332  005737  001362          TST     SEC3D           ;FORMAT IN 16 BIT MODE
011336  100402                  BMI     3$              ;BRANCH IF IT IS
011340  012702  051630          MOV     #BUFP+4+<10.*516.>,R2 ;ECTOR 10
011344  012703  004430  3$:     MOV     #USSAV,R3       ;LOAD THE TABLE
011350  012704  000400          MOV     #256.,R4        ;WORD COUNT
011354  012223          4$:     MOV     (R2)+,(R3)+     ;
011356  005304                  DEC     R4
011360  001375                  BNE     4$
011362  000240          5$:     NOP                     ;DONE
2277
2278                            ;THIS CODE ASK O.P TO ENTER A SERIAL NUMBER FOR UN FORMATTED PACK
2279                            ;
2280                            ;
2281
011364  005737  001406  SERNL:  TST     PACK            ;BAD SPOT FILE AVAILABLE ?
011370  003124                  BGT     4$              ;BRANCH IF AVAILABLE
011372  105737  001150          TSTB    $AUTOB          ;RUN UNDER APT ?
011376  001144                  BNE     5$              ;BRANCH IF IT IS
011400  004737  021772          JSR     PC,$TKINT       ;INITIALTHE KEYBOARD
011404  104401  001203          TYPE    ,$CRLF          ;TYPE CRLF
011410  104401  033777          TYPE    ,MESG1          ;"ENTER TWO WORDS FOR SN"
011414  104401  034065  1$:     TYPE    ,SN1            ;
011420  012702  000012          MOV     #10.,R2         ;MAXMUM 10 OCTAL DIGITS ALLOWED
011424  005037  001414          CLR     BAD16           ;CLEAR THE LSW OF THE SERIAL NUMBER
011430  005037  001416          CLR     BAD16+2         ;CLEAR THE MSW OF THE SERIAL NUMBER
011434  104411                  RDLIN                   ;READ IN THE STRING
011436  012601                  MOV     (SP)+,R1        ;ADDRESS OF THE READ IN STRING
011440  105711          2$:     TSTB    (R1)            ;TERMINATOR OF THE INPUT STRING
011442  001471                  BEQ     3$              ;BRANCH IF IT IS
011444  121127  000060          CMPB    (R1),#'0        ;LESS THAN ASCIZ 0 ?
011450  103761                  BLO     1$              ;ENTER AGAIN IF IT IS
011452  121127  000067          CMPB    (R1),#'7        ;HIGH THAN ASCIX 7 ?
011456  101356                  BHI     1$              ;ENTER AGAIN IF SO
011460  013746  001414          MOV     BAD16,-(SP)     ;SAVE THE LSW
011464  042737  100000  001414  BIC     #BIT15,BAD16    ;CLEAR THE SIGN BIT OF LSW
011472  006137  001414          ROL     BAD16           ;ROTATE THE LSW FIVE TIMES
011476  006137  001414          ROL     BAD16           ;TO MOVE THE MOST SIGN DIGIT
011502  006137  001414          ROL     BAD16           ;TO BIT 0 TO BIT 2
011506  006137  001414          ROL     BAD16           ;
011512  006137  001414          ROL     BAD16           ;
011516  042737  177770  001414  BIC     #177770,BAD16   ;LEFT ON THE MS DIGIT OF THE LSW
011524  006337  001416          ASL     BAD16+2         ;SHIFT THE MSW  LEFT ONE OCTAL DIGIT
011530  100731                  BMI     1$              ;ENTER AGAIN IF SIGN BIT SET
011532  006337  001416          ASL     BAD16+2         ;
011536  100726                  BMI     1$              ;
011540  006337  001416          ASL     BAD16+2         ;
011544  100723                  BMI     1$              ;
011546  063737  001414  001416  ADD     BAD16,BAD16+2   ;APPENDING THE DIGITS INTO MSW
011554  012637  001414          MOV     (SP)+,BAD16     ;RESTORE THE LSW
011560  006337  001414          ASL     BAD16           ;SHIFT THE LSW LEFT ONE OCTAL DIGIT
011564  006337  001414          ASL     BAD16           ;
011570  006337  001414          ASL     BAD16           ;
011574  042737  100000  001414  BIC     #BIT15,BAD16    ;CLEAR THE SIGN BIT
011602  111103                  MOVB    (R1),R3         ;LOAD THE CURRENT READ IN DIGITS
```

```
2322  011604  042703  177770                      BIC     #177770,R3      ;LEFT ONLY ONE OCTAL DIGIT
2323  011610  060337  001414                      ADD     R3,BAD16        ;APPEND THE READ IN CHARACTER TO LSW
2324  011614  005201                              INC     R1              ;ADJUST THE READ IN STRING ADDRESS
2325  011616  005302                              DEC     R2              ;DECREMENT ONE DIGIT COUNT
2326  011620  001307                              BNE     2$              ;BRANCH IF NOT DONE
2327  011622  105711                              TSTB    (R1)            ;IF 10 DIGITS HAVE BEEN ENTERED
2328  011624  001273                              BNE     1$              ;MUST BE FOLLOWED BY <CR>
2329  011626  005737  001414              3$:     TST     BAD16           ;SERIAL NUMBER MUST BE NOT 0
2330  011632  001003                              BNE     4$              ;BRANCH IF NOT 0
2331  011634  005737  001416                      TST     BAD16+2         ;
2332  011640  001665                              BEQ     1$              ;ENTER AGAIN IF ZERO
2333  011642  013737  001414  002420      4$:     MOV     BAD16,BAD18             ;LOAD ALL TABLE WITH THE SERIAL NUMBER
2334  011650  013737  001414  003424              MOV     BAD16,USTAB             ;
2335  011656  013737  001414  004430              MOV     BAD16,USSAV             ;
2336  011664  013737  001416  003426              MOV     BAD16+2,USTAB+2 ;
2337  011672  013737  001416  004432              MOV     BAD16+2,USSAV+2 ;
2338  011700  013737  001416  002422              MOV     BAD16+2,BAD18+2 ;
2339  011706  000410                              BR      6$              ;EXIT
2340  011710  005237  004430              5$:     INC     USSAV           ;CODE FOR AUTO MODE APT,ACT,ETC.
2341  011714  005237  001414                      INC     BAD16
2342  011720  005237  002420                      INC     BAD18
2343  011724  005237  003424                      INC     USTAB
2344  011730  000240                      6$:     NOP                     ;DONE
2345  011732  012737  016344  000060              MOV     #NEWSVC,@#TKVEC ;NEW INTERRUPT VECTOR FOR KEYBOARD
2346  011740  005037  177776                      CLR     @#PSW
2347                                  ;SETUP TOTAL TRACK COUNT FOR FORMAT
2348
2349  011744  023737  001320  001322  CKADRS: CMP     ENDCYL,BEGCYL   ;STARTING AND ENDING CYLINDERS THE SAME ?
2350  011752  001011                              BNE     5$              ;BRANCH IF BEGCYL NOT EQUAL TO ENDCYL
2351  011754  013737  001324  001330              MOV     ENDTRK,TTRKS    ;END TRACK ADDRESS
2352  011762  163737  001326  001330              SUB     BEGTRK,TTRKS    ;SUBTRACT THE STARTING TRACK ADDRESS
2353  011770  005237  001330                      INC     TTRKS           ;MAKE THE NUMBER OF TRACKS INCLUSIVE
2354  011774  000450                              BR      SETPAT          ;SETUP THE DATA PATTERN
2355  011776  005737  001402              5$:     TST     WRAP            ;WRAP AROUND FLAG SET ?
2356  012002  001407                              BEQ     1$              ;NO
2357  012004  012702  001466                      MOV     #822.,R2        ;INITIAL COUNTER TO 822
2358  012010  163702  001322                      SUB     BEGCYL,R2       ;FIRST PART OF CYL #
2359  012014  063702  001320                      ADD     ENDCYL,R2       ;SECOND PART OF CYL #
2360  012020  000405                              BR      4$
2361  012022  013702  001320              1$:     MOV     ENDCYL,R2       ;ENDING CYLINDER
2362  012026  163702  001322                      SUB     BEGCYL,R2       ;SUBTRACT THE STARTING CYLINDER
2363  012032  005302                              DEC     R2              ;EXCLUSIVE LAST CYLINDER
2364  012034  012737  000005  001330      4$:     MOV     #5,TTRKS        ;INITATE COUNTER
2365  012042  163737  001326  001330              SUB     BEGTRK,TTRKS    ;# OF TRACK ON STARTING CYLINDER
2366  012050  063737  001324  001330              ADD     ENDTRK,TTRKS    ;# OF TRACK ON ENDING CYLINDER
2367  012056  005237  001330                      INC     TTRKS           ;INCLUSIVE ONE TRACK
2368  012062  005302                      3$:     DEC     R2              ;DECREMENT THE CYLINDER COUNT
2369  012064  100404                              BMI     6$              ;BR IF DONE
2370  012066  062737  000005  001330              ADD     #5.,TTRKS       ;ADD CYLINDER WORTH OF TRACKS
2371  012074  000772                              BR      3$              ;CONTINUE
2372  012076  005037  001410              6$:     CLR     EDIT            ;RESET EDIT FLAG
2373  012102  023727  001330  010011              CMP     TTRKS,#<821.*5> ;ALL TRACKS ARE FORMATTED ?
2374  012110  103402                              BLO     SETPAT          ;BRANCH IF NOT
2375  012112  005237  001410                      INC     EDIT            ;SET EDIT FLAG
2376
2377                                  ;THIS CODE SETS UP THE SELECTED DATA PATTERN TO BE WRITTEN ON EACH SECTOR IMAGE
```

```
2378
2379  012116  005737  001316          SETPAT: TST     MODE            ;VERIFY MODE ?
2380  012122  001427                          BEQ     1$              ;YES
2381  012124  005037  001340          4$:     CLR     PATA            ;CLEAR DATA PATTERN A
2382  012130  005037  001342                  CLR     PATB            ;CLEAR DATA PATTERN B
2383  012134  005737  001336                  TST     PATSEL          ;SEE IF PATTERN OF ONES
2384  012140  001420                          BEQ     1$              ;BR IF SO
2385  012142  012737  177777  001340          MOV     #177777,PATA    ;PATA = AB
2386  012150  012737  177777  001342          MOV     #177777,PATB    ;PATB=CD
2387  012156  022737  000001  001336          CMP     #1,PATSEL       ;SEE IF PATTERN OF ZEROS
2388  012164  001406                          BEQ     1$              ;BRANCH IF SO
2389  012166  012737  066666  001340          MOV     #066666,PATA    ;SET UP WORST CASE
2390  012174  012737  066666  001342          MOV     #066666,PATB    ;SET UP WORST CASE
2391  012202  013703  001354          1$:     MOV     WC,R3           ;SET UP COUNTER
2392  012206  012700  037554                  MOV     #BUFP,R0        ;SET UP MEMORY POINTER
2393  012212  013701  001340          3$:     MOV     PATA,R1         ;SET UP PATTERN IN R1
2394  012216  013702  001342                  MOV     PATB,R2         ;SET UP PATTERN IN R2
2395  012222  010120                  2$:     MOV     R1,(R0)+        ;MOV 1ST PAT INTO MEM
2396  012224  010220                          MOV     R2,(R0)+        ;MOV 2ND PAT INTO MEM
2397  012226  005303                          DEC     R3              ;KEEP COUNT
2398  012230  005303                          DEC     R3              ;KEEP COUNT
2399  012232  001373                          BNE     2$              ;DO IT AGAIN IF R3 NOT 0
2400                          ;THIS CODE SETS UP FOR THE ACTUAL FORMAT
2401
2402  012234  004737  015376          WRTRK:  JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
2403  012240  004737  015710                  JSR     PC,SETHDR       ;GO INITIALIZE HEADERS IN THE SECTOR BUFFER IN CORE
2404  012244  112737  000020  005611          MOVB    #20,FMTDPB+1    ;LOAD FMT22 BIT
2405  012252  005737  001362                  TST     SEC30           ;18 BIT MODE ?
2406  012256  001002                          BNE     1$              ;BR IF NOT
2407  012260  105037  005611                  CLRB    FMTDPB+1        ;CLEAR THE FMT22 BIT
2408  012264  112737  000143  005612  1$:     MOVB    #SETFMT,FMTDPB+2  ;'LOAD FORMAT' COMMAND
2409  012272  004037  024736          2$:     JSR     R0,RM03         ;START THE COMMAND
2410  012276  005610                          FMTDPB                  ;DPB ADDRESS
2411  012300  000774                          BR      2$              ;QUEUE FULL RETURN
2412  012302  005737  005626          3$:     TST     FMTDPB+16       ;LOOK FOR DONE
2413  012306  001775                          BEQ     3$              ;LOOP UNTIL FINISHED
2414
2415                          ;SET UP SECCU TABLE: BAD SPOT ON CURRENT TRACK ,RETRIEVED FROM PACK
2416                          ;
2417  012310  012703  005434          WRTRKX: MOV     #SECCU,R3       ;INITILIZE THE SECCU TABLE
2418  012314  012704  000040                  MOV     #32.,R4         ;TOTAL 32 WORD MAX
2419  012320  012702  177777                  MOV     #-1,R2          ;LOAD ALL LOCATION TO -1
2420  012324  010223                  1$:     MOV     R2,(R3)+        ;
2421  012326  005304                          DEC     R4
2422  012330  001375                          BNE     1$              ;BRANCH IF NOT DONE
2423                          ;
2424                          ;LOAD THE BAD SPOT INTO SECCU TABLE IF ANY
2425                          ;
2426  012332  005737  001406          WRTRKY: TST     PACK            ;BAD SPOT FILE AVAILABLE AT ALL
2427  012336  003451                          BLE     5$              ;BRANCH IF NOT
2428  012340  012702  001414                  MOV     #BAD16,R2       ;R2 POINTER OF MFG BAD SPOT TABLE
2429  012344  005737  001362                  TST     SEC30           ;IF 18 BIT MODE CHANGE POINTER
2430  012350  100402                          BMI     1$              ;BRANCH IF 16 BIT MODE
2431  012352  012702  002420                  MOV     #BAD18,R2       ;
2432  012356  010201                  1$:     MOV     R2,R1           ;R1: LAST ADDRESS OF BAD SPOT TABLE
2433  012360  062701  000776                  ADD     #510.,R1        ;
```

```
2434  012364  012703  005434                    MOV     #SECCU,R3       ;TABLE ADDRESS
2435  012370  012704  000076                    MOV     #62.,R4         ;LAST ADDRESS OF THE TABLE
2436  012374  060304                            ADD     R3,R4           ;
2437  012376  062702  000010                    ADD     #10,R2          ;BAD SPOT STARTS FROM THE 5TH WORD OF THE FILE
2438  012402  023712  005622            2$:     CMP     FMTDPB+12,(R2)  ;BAD SPOT ON THIS CYLINDER ?
2439  012406  001007                            BNE     3$              ;BRANCH IF NOT
2440  012410  123762  005621  000003            CMPB    FMTDPB+11,3(R2) ;ALSO, ON THE SAME TRK ?
2441  012416  001003                            BNE     3$              ;BRANCH IF NOT
2442  012420  116205  000002                    MOVB    2(R2),R5        ;BAD SECTOR
2443  012424  010523                            MOV     R5,(R3)+        ;WORD STORAGE
2444  012426  062702  000004            3$:     ADD     #4,R2           ;ADVANCE TO NEXT SET
2445  012432  020102                            CMP     R1,R2           ;ALL SPOT ARE CHECKED ?
2446  012434  103412                            BLO     5$              ;BRANCH IF IT IS
2447  012436  020403                            CMP     R4,R3           ;END OF SECCU TABLE ?
2448  012440  103401                            BLO     4$              ;REJECT THE PACK
2449  012442  000757                            BR      2$              ;LOOPING BACK
2450  012444  004737  016246            4$:     JSR     PC,RESTRT               ;IMPORPER MFG INFORMATION
2451  012450  012737  177777  001406            MOV     #-1,PACK                ;RESET MFG FILE AVAILABLE FLAG
2452  012456  000137  011172                    JMP     TABINT                  ;AND START AGAIN
2453  012462  000240                    5$:     NOP                     ;DONE
2454
2455                                    ;THIS CODE SORTS THE BAD SECTOR TABLE "SECCU" IN ASCENGING ORDER
2456                                    ;AND DELETES THE DUPLICATED ONES
2457                                    ;
2458  012464  012702  005434            SORT1:  MOV     #SECCU,R2       ;TOP OF THE TABLE
2459  012470  012703  005532                    MOV     #SECCU+62.,R3   ;BOTTOM OF THE TABLE
2460  012474  012704  000037                    MOV     #31.,R4 ;TOTAL 32 ELEMENT,SORT 31 TIMES
2461  012500  022712  177777                    CMP     #-1,(R2)        ;IS THE TABLE EMPTY ?
2462  012504  001451                            BEQ     6$              ;BRANCH IF IT IS, QUICK EXIT
2463  012506  021262  000002            1$:     CMP     (R2),2(R2)      ;N TH ELEMENT : N+1 TH ELEMENT
2464  012512  103430                            BLO     4$              ;BRANCH IF SMALLER
2465  012514  001406                            BEQ     2$              ;DELET N+1 TH ELEMENT, IF SAME
2466                                                                    ;OTHERWISE, N > N+1
2467  012516  011246                            MOV     (R2),-(SP)      ;STORE N ELEMENT
2468  012520  016212  000002                    MOV     2(R2),(R2)      ;SWITCH N+1 ELEMENT TO N ELEMENT
2469  012524  012662  000002                    MOV     (SP)+,2(R2)     ;SWITCH N ELEMENT TO N+1 ELEMENT
2470  012530  000421                            BR      4$              ;ADJUST TOP POINTER
2471  012532  022712  177777            2$:     CMP     #-1,(R2)        ;TERMINATOR ?
2472  012536  001416                            BEQ     4$              ;BRANCH IF IT IS
2473  012540  010246                            MOV     R2,-(SP)        ;SAVE TOP POINTER
2474  012542  062702  000002                    ADD     #2,R2           ;DELET 2(R2)
2475  012546  016222  000002            3$:     MOV     2(R2),(R2)+     ;
2476  012552  022702  005532                    CMP     #SECCU+62.,R2   ;BOTTOM OF THE TABLE ?
2477  012556  101373                            BHI     3$              ;BRANCH IF NOT
2478  012560  012737  177777  005532            MOV     #-1,SECCU+62.   ;LOAD -1 TO TABLE BOTTOM
2479  012566  012602                            MOV     (SP)+,R2        ;RESTORE THE TOP POINTER
2480  012570  005304                            DEC     R4              ;DECREMENT ONE ELEMENT COUNT
2481  012572  001416                            BEQ     6$              ;BRANCH IF ALL DONE
2482  012574  062702  000002            4$:     ADD     #2,R2           ;INCREMENT THE TOP POINTER
2483  012600  020302                            CMP     R3,R2           ;REACH THE BOTTOM ?
2484  012602  001341                            BNE     1$              ;BRANCH IF NOT
2485  012604  005304                    5$:     DEC     R4              ;DECREMENT ONE SORT COUNT
2486  012606  001410                            BEQ     6$              ;BRANCH IF ALL DONE
2487  012610  012702  005434                    MOV     #SECCU,R2       ;RESET TOP POINTER
2488  012614  162703  000002                    SUB     #2,R3           ;UPDATE BOTTOM POINTER
2489  012620  022703  005434                    CMP     #SECCU,R3       ;EXIT IF TOP = BOTTOM
```

```
2490  012624  001401                          BEQ     6$              ;
2491  012626  000727                          BR      1$              ;SORT THE REST ELEMENTS
2492  012630  000240                  6$:     NOP                     ;DONE
2493  012632  005037  005536                  CLR     NEXT            ;FLAG,START OF A NEW TRACK
2494  012636  005037  001314          WRTRK1: CLR     SOFSW           ;CLEAR ERROR COUNTER
2495  012642  005037  001346                  CLR     RETRY           ;ZERO THE RETRY COUNTER
2496  012646  105037  005620                  CLRB    FMTDPB+10       ;RESTORE SECTOR
2497  012652  013737  001356  005614          MOV     MWC,FMTDPB+4    ;RESTORE WC
2498  012660  012737  037554  005616          MOV     #BUFP,FMTDPB+6  ;RESTORE BA
2499                          ;DELET THE FOLLOWING TWO LINES FOR NEW VERIFY OPERATION
2500                          ;       TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
2501                          ;       BEQ     CKTRK           ;BR IF 'CHECK' MODE
2502                          ;
2503                          ;THIS CODE TO SET UP WORD COUNT AND BUFFER ADDRESS
2504                          ;
2505  012666  022737  177777  005536  WRTRKA: CMP     #-1,NEXT        ;END OF OPERATION ?
2506  012674  001001                          BNE     1$              ;BRANCH IF NOT
2507  012676  000000                          HALT                    ;OTHERWISE SOMETHING WRONG *******
2508  012700  113737  005536  005620  1$:     MOVB    NEXT,FMTDPB+10  ;STARTING SECTOR
2509  012706  012701  005434                  MOV     #SECCU,R1       ;R1 POINTS TO THE BAD SECTOR TABLE
2510  012712  022711  177777          2$:     CMP     #-1,(R1)        ;END OF TABLE ?
2511  012716  001505                          BEQ     WRTRKD          ;BRANCH IF IT IS
2512  012720  032711  100000                  BIT     #BIT15,(R1)     ;THE BAD SECTOR HAS ALREADY ACCESSED ?
2513  012724  001403                          BEQ     3$              ;BRANCH IF NOT
2514  012726  062701  000002                  ADD     #2,R1           ;ADJUST THE POINTER
2515  012732  000767                          BR      2$              ;LOOPING BACK
2516  012734  123711  005620          3$:     CMPB    FMTDPB+10,(R1)  ;BAD SPOT = STARTING SECTOR ?
2517  012740  001414                          BEQ     WRTRKB          ;BRANCH , IF IT I
2518  012742  103401                          BLO     4$              ;BRANCH IF START SEC < BAD SPOT
2519  012744  000000                          HALT                    ;OTHERWISE , SOMETHING WRONG
2520  012746  111102                  4$:     MOVB    (R1),R2         ;UPDATE NEXT STARTING ADDRESS
2521  012750  023702  001364                  CMP     MAXSEC,R2       ;NEXT BAD SPOT IS THE LAST SECTOR?
2522  012754  101466                          BLOS    WRTRKD          ;BRANCH IF IT IS
2523  012756  005202                          INC     R2              ;NEXT = BAD + 1
2524  012760  010237  005536                  MOV     R2,NEXT         ;NEXT STARTING ADDRESS
2525  012764  052721  100000                  BIS     #BIT15,(R1)+    ;MARK THE ACCESSED SECTOR
2526  012770  000435                          BR      WRTRKC          ;TO SET UP WORD COUNT
2527  012772  105237  005620  WRTRKB: INCB    FMTDPB+10               ;INCREMENT THE STARTING SECTOR
2528  012776  052721  100000                  BIS     #BIT15,(R1)+    ;MARK THE ACCESSED SECTOR
2529  013002  123711  005620                  CMPB    FMTDPB+10,(R1)  ;STILL ON ANOTHER BAD SPOT ?
2530  013006  001771                          BEQ     WRTRKB                  ;LOOP AGAIN
2531  013010  123737  005620  001364          CMPB    FMTDPB+10,MAXSEC        ;IF START SECTOR > MAXSEC
2532  013016  101407                          BLOS    1$              ;IMPORPER MFG INDORMATION
2533  013020  004737  016246                  JSR     PC,RESTRT       ;START AGAIN AND REBUILD FMG FILE
2534  013024  012737  177777  001406          MOV     #-1,PACK        ;RESET THE MFG FILE AVAILABLE FLAG
2535  013032  000137  011172                  JMP     TABINT          ;RESTART POINT
2536  013036  022711  177777          1$:     CMP     #-1,(R1)        ;TO SEE IF NEXT BAD SPOT IN TABLE
2537  013042  001433                          BEQ     WRTRKD          ;BRANCH IF NONE
2538  013044  111102                          MOVB    (R1),R2         ;UPDATE THE NEXT
2539  013046  023702  001364                  CMP     MAXSEC,R2       ;BAD SECTOR IS THE LAST SECTOR ?
2540  013052  101427                          BLOS    WRTRKD          ;BRANCH IF IT IS
2541  013054  005202                          INC     R2              ;BAD SEC +1
2542  013056  010237  005536                  MOV     R2,NEXT         ;NEXT STARTING ADDRESS
2543  013062  052711  100000                  BIS     #BIT15,(R1)     ;MARK THE ACCESSED SECTOR
2544  013066  013702  005536  WRTRKC: MOV     NEXT,R2         ;CALCULATE THE WORD COUNT
2545  013072  162702  000002                  SUB     #2,R2           ;ENDSECTOR = NEXT - 2
```

```
2546  013076  006302                        ASL     R2              ;WORD INDEX
2547  013100  016203  006052                MOV     WCTBL(R2),R3    ; WORD CTR FOR SECTOR 0 TO ENDING SECTOR
2548  013104  113702  005620                MOVB    FMTDPB+10,R2    ;STARTING SECTOR
2549  013110  006302                        ASL     R2              ;WORD INDEX
2550  013112  166203  006052                SUB     WCTBL(R2),R3    ;WORD COUNT FOR SECTOR 0 TO STARTING SECTOR
2551  013116  062703  000402                ADD     #258.,R3                ;ADJUST ONE SECTOR
2552  013122  005403                        NEG     R3              ;GET THE NEAGTIVE WORD COUNT
2553  013124  010337  005614                MOV     R3,FMTDPB+4     ;LOAD THE WORD CTR INTO DPB
2554  013130  000423                        BR      WRTRKE          ;LOAD THE STARTING ADDRESS
2555  013132  012737  177777  005536  WRTRKD: MOV     #-1,NEXT        ;NO MORE BAD SPOT
2556  013140  013702  001364                MOV     MAXSEC,R2       ;LAST SECTOR
2557  013144  006302                        ASL     R2              ;WORD INDEX
2558  013146  016203  006052                MOV     WCTBL(R2),R3    ;WORD CTR FOR THE ENDING SECTOR
2559  013152  113702  005620                MOVB    FMTDPB+10,R2    ;STARTING ADDRESS
2560  013156  006302                        ASL     R2              ;WORD INDEX
2561  013160  166203  006052                SUB     WCTBL(R2),R3    ;WORD CTR FOR THE STARTING SECTOR
2562  013164  062703  000402                ADD     #258.,R3        ;ADJUST ONE SECTOR
2563  013170  005403                        NEG     R3              ;NEGATIVE WORD COUNT
2564  013172  010337  005614                MOV     R3,FMTDPB+4     ;LOAD THE WORD COUNT INTO DPB
2565  013176  000240                        NOP                     ;DONE
2566  013200  113702  005620          WRTRKE: MOVB    FMTDPB+10,R2    ;STARTING ADDRESS
2567  013204  006302                        ASL     R2              ;LOCATE THE BUFFER ADDRESS
2568  013206  016237  005752  005616        MOV     ADRTBL(R2),FMTDPB+6     ;BUFFER ADDRESS
2569  013214  112737  000163  005612  WRTRK2: MOVB    #WRTHD,FMTDPB+2 ;SET WRITE HEADER & DATA COMMAND IN TBL
2570  013222  012737  013222  001124        MOV     #.,$LPERR       ;SET UP LOOP ON ERROR ADDRESS
2571  013230  012706  001100                MOV     #STACK,SP       ;LOAD STACK POINT
2572  013234  004037  024736          1$:   JSR     R0,RM03         ;GO FORMAT A TRACK
2573  013240  005610                        FMTDPB                  ;ADRS OF PARAMETERS - TBL
2574  013242  000774                        BR      1$              ;WAIT FOR QUEUE IF FULL
2575  013244  005737  005626          2$:   TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
2576  013250  001775                        BEQ     2$              ;BRANCH IF NOT DONE
2577  013252  100024                        BPL     4$              ;BRANCH IF NO ERROR
2578  013254  012737  013302  001174        MOV     #3$,$ESCAPE     ;ESCAPE TO 3$ ON ERROR
2579  013262  004737  015230                JSR     PC,ERINDX       ;SEE WHICH ERROR
2580  013266  104010                        ERROR   10              ;DRIVE OFFLINE
2581  013270  104011                        ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
2582  013272  104012                        ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
2583  013274  104013                        ERROR   13              ;SOFTWARE TIMEOUT
2584  013276  104014                        ERROR   14              ;DRIVE UNSAFE ERROR
2585  013300  104015                        ERROR   15              ;DRIVE/CONTROLLER ERROR DURING WRITE
2586  013302  004737  015330          3$:   JSR     PC,LOP.CK       ;LOOP ON THE ERROR ?
2587  013306  022737  000003  001346        CMP     #3,RETRY        ;ERROR RETRY LIMIT ?
2588  013314  001403                        BEQ     4$              ;BR IF REACHED
2589  013316  005237  001346                INC     RETRY           ;COUNT THE ERROR
2590  013322  000744                        BR      1$              ;TRY AGAIN
2591  013324  005037  001174          4$:   CLR     $ESCAPE         ;CLEAR ERROR ESCAPE ADDRESS
2592  013330  005037  001346                CLR     RETRY           ;CLEAR THE RETRY COUNTER
2593
2594                                  ;CHECK THE TRACK JUST WRITTEN
2595
2596  013334  112737  000153  005612  CKTRK:  MOVB    #WCKHD,FMTDPB+2 ;SET WRITE CHECK HEADER & DATA COMMAND IN TBL
2597  013342  012737  013342  001124        MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
2598  013350  012706  001100                MOV     #STACK,SP       ;LOAD STACK POINTER
2599  013354  004037  024736          1$:   JSR     R0,RM03         ;GO CHECK THE TRACK JUST FORMATTED
2600  013360  005610                        FMTDPB                  ;ADRS OF PARAMETERS - TBL
2601  013362  000774                        BR      1$              ;WAIT FOR QUEUE IF FULL
```

```
2602  013364  005737  005626          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
2603  013370  001775                          BEQ     2$              ;BRANCH IF NOT DONE
2604  013372  100131                          BPL     8$              ;BRANCH IF NO ERROR
2605  013374  113737  005546  001350          MOVB    RM.REG+RMDA,SAVSEC  ;GET THE SECTOR ADDRESS
2606  013402  001005                          BNE     3$              ;BRANCH IF NOT SECTOR 0
2607  013404  013737  001364  001350          MOV     MAXSEC,SAVSEC   ;RESTORE TO LAST SECTOR +1
2608  013412  005237  001350                  INC     SAVSEC          ;INCREMENT TO LAST SECTOR +1
2609  013416  005337  001350          3$:     DEC     SAVSEC          ;ADJUST SECTOR TO THE ONE THAT FAILED
2610  013422  012737  013724  001174          MOV     #10$,$ESCAPE    ;ESCAPE TO 10$ ON ERROR
2611  013430  004737  015230                  JSR     PC,ERINDX       ;SEE WHICH ERROR
2612  013434  104010                          ERROR   10              ;DRIVE OFFLINE
2613  013436  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
2614  013440  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
2615  013442  104013                          ERROR   13              ;SOFTWARE TIMEOUT
2616  013444  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
2617  013446  032737  040000  005550          BIT     #WCE,RM.REG+RMCS2   ;WRITE CHECK ERROR ?
2618  013454  001005                          BNE     4$              ;BR IF SET
2619  013456  033727  005554                  BIT     RM.REG+RMER1,(PC)+  ;CHECK FOR DATA ERRORS
2620  013462  130620                          .WORD   DCK!OPI!DTE!HCRC!HCE!FER  ;DATA ERROR BITS
2621  013464  001001                          BNE     4$              ;BR IF SET
2622  013466  104016                          ERROR   16              ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
2623  013470  005737  001314          4$:     TST     SOFSW           ;RETRYING THE SECTOR ?
2624  013474  001413                          BEQ     5$              ;BR IF NOT
2625  013476  012737  013750  001174          MOV     #11$,$ESCAPE    ;ESCAPE TO 11$ ON ERROR
2626  013504  004737  016616                  JSR     PC,RECORD       ;RECORD THE BAD SPOT
2627  013510  012737  100000  001412          MOV     #BIT15,HEADX    ;RESET THE USER BIT
2628  013516  004737  017436                  JSR     PC,RESET        ;RESET THE HEADER BITS
2629  013522  104017                          ERROR   17              ;SECTOR NOT ACCEPTABLE
2630  013524  005037  001174          5$:     CLR     $ESCAPE         ;RESET ESCAPE ADDRESS
2631  013530  032737  040000  005552          BIT     #ERR,RM.REG+RMDS    ;DATA ERROR ?
2632  013536  001002                          BNE     6$              ;BR IF IT IS
2633  013540  104024                          ERROR   24              ;WRITE CHECK ERROR
2634  013542  000401                          BR      7$              ;CONTINUE
2635  013544  104020                  6$:     ERROR   20              ;DATA ERROR DURING WRITE CHECK
2636  013546  013701  001350          7$:     MOV     SAVSEC,R1       ;FAILING SECTOR ADDRESS
2637  013552  113737  001350  005620          MOVB    SAVSEC,FMTDPB+10    ;SECTOR ADDRESS
2638  013560  006301                          ASL     R1              ;SETUP INDEX TO ADDRESS WORDS
2639  013562  016137  005752  005616          MOV     ADRTBL(R1),FMTDPB+6  ;BUFFER ADDRESS FOR FAILING SECTOR
2640  013570  013701  001364                  MOV     MAXSEC,R1       ;R1 ENDING SECTOR #
2641  013574  005737  005536                  TST     NEXT            ;WHOLE TRACK DONE ?
2642  013600  100404                          BMI     13$             ;BRANCH IF IT IS
2643  013602  013701  005536                  MOV     NEXT,R1         ;LOAD CURRENT ENDING SECTOR
2644  013606  162701  000002                  SUB     #2,R1           ;ADJUST TWO SECTORS
2645  013612  006301                  13$:    ASL     R1              ;WORD INDEX
2646  013614  016102  006052                  MOV     WCTBL(R1),R2    ;WORDCTR FOR ENDING SECTOR
2647  013620  113701  005620                  MOVB    FMTDPB+10,R1    ;STARTING ADDRESS
2648  013624  006301                          ASL     R1              ;WORD COUNT
2649  013626  166102  006052                  SUB     WCTBL(R1),R2    ;WORD CTR FOR STARTING SECTOR
2650  013632  005402                          NEG     R2
2651  013634  010237  001352                  MOV     R2,SAVWC        ;SAVE THE WORD COUNT
2652  013640  012737  177376  005614          MOV     #-258.,FMTDPB+4 ;WORD COUNT FOR 1 SECTOR
2653  013646  005237  001314                  INC     SOFSW           ;INDICATE THAT A RETRY IS IN PROGRESS
2654  013652  000137  013214                  JMP     WRTRK2          ;REFORMAT ERROR SECTOR
2655  013656  005737  001314          8$:     TST     SOFSW           ;RETRY IN PROGRESS ?
2656  013662  001432                          BEQ     11$             ;BR IF NOT
2657  013664  022737  000002  001314          CMP     #2,SOFSW        ;SEE IF LAST RETRY
```

```
2658  013672  001403                        BEQ     9$              ;BR IF IT IS
2659  013674  005237  001314                INC     SOFSW           ;INCREMENT RETRY COUNT
2660  013700  000625                        BR      1$              ;READ AGAIN
2661  013702  123737  001364  005620  9$:   CMPB    MAXSEC,FMTDPB+10' ;SEE IF LAST SECTOR ON THE TRACK
2662  013710  001417                        BEQ     11$             ;BR IF IT IS
2663  013712  004737  015662                JSR     PC,SCAWC        ;SETUP TO CHECK REMAINING SECTORS ON THE TRACK
2664  013716  005037  001314                CLR     SOFSW           ;CLEAR RETRY COUNTER
2665  013722  000614                        BR      1$              ;FINISH CHECKING THE TRACK
2666  013724  004737  015330          10$:  JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
2667  013730  022737  000003  001346        CMP     #3,RETRY        ;ERROR RETRY REACHED ?
2668  013736  001404                        BEQ     11$             ;BR IF IT IS
2669  013740  005237  001346                INC     RETRY           ;COUNT THE RETRY
2670  013744  000137  013354                JMP     1$              ;DO THE WRITE CHECK AGAIN
2671  013750  005037  001174          11$:  CLR     $ESCAPE         ;CLEAR THE ERROR RETURN ESCAPE ADDRESS
2672  013754  005037  001346                CLR     RETRY           ;CLEAR THE RETRY COUNTER
2673  013760  005037  001314                CLR     SOFSW           ;CLEAR THE SOFTWARE RETRY COUNT
2674  013764  005737  005536                TST     NEXT            ;WHOLE TRACK DONE ?
2675  013770  100402                        BMI     .+6             ;BRANCH IF IT IS
2676  013772  000137  012666                JMP     WRTRKA
2677  013776  032777  000002  165150        BIT     #BIT1,@SWR      ;LOOP ON CURRENT TRACK ?
2678  014004  001402                        BEQ     12$             ;BR IF NOT
2679  014006  000137  012310                JMP     WRTRKX          ;START AGAIN ON THE SAME TRACK
2680                                                                ;RESET THE HEADER WORD OF ALL MFG
2681                                                                ;DEFINED BAD SECTOR
2682  014012  012737  040000  001412  12$:   MOV     #BIT14,HEADX    ;RESET THE MFG BIT OF THE 1ST HEADER WORD
2683  014020  012704  005434                MOV     #SECCU,R4       ;TABLE FOR BAD SECTOR ON CURRENT TRACK
2684  014024  022714  177777          14$:  CMP     #-1,(R4)        ;END OF TABLE ?
2685  014030  001414                        BEQ     16$             ;BRANCH IF SO
2686  014032  032714  040000                BIT     #BIT14,(R4)     ;HAS THE SECTOR BE ASSCESSED ?
2687  014036  001006                        BNE     15$             ;BRANCH IF SO
2688  014040  052714  040000                BIS     #BIT14,(R4)     ;MASK THE SECTOR TABLE
2689  014044  111437  001350                MOVB    (R4),SAVSEC     ;SECTOR ADDRESS
2690  014050  004737  017436                JSR     PC,RESET        ;RESET THE HEADER
2691  014054  062704  000002          15$:  ADD     #2,R4           ;INCREMENT THE TABLE POINTER
2692  014060  000761                        BR      14$             ;LOOPING BACK
2693  014062  000240          16$:  NOP                             ;ALL DONE
2694
2695  014064  004737  015466          WRTRKZ: JSR   PC,TRKTST       ;GET UPDATED TRACK AND CYLINDER ADDRESSES
2696  014070  000402                        BR      HDREAD          ;RETURN HERE IF DONE - DO QUICK CHECK OF DISK
2697  014072  000137  012310                JMP     WRTRKX          ;CONTINUE WITH THE FORMAT
2698
2699                                        ;THIS CODE MAKES SURE EACH CYLINDER FORMATTED CONTAINS THE
2700                                        ;PROPER CYLINDER ADDRESS. THE PROGRAM IS LOOKING FOR
2701                                        ;POSSIBLE POSITIONER ERRORS THAT MAY HAVE OCCURRED DURING THE FORMAT.
2702
2703  014076  005737  001316          HDREAD: TST   MODE            ;VERIFY MODE ?
2704  014102  001020                        BNE     9$              ;NO
2705  014104  022737  066666  001340        CMP     #066666,PATA    ;PATTERN HAS BEEN ROTATED TO WORST CASE ?
2706  014112  001575                        BEQ     $EOP            ;YES
2707  014114  000241                        CLC                     ;CLEAR THE CARRY BIT
2708  014116  006037  001340                ROR     PATA            ;ROTATE PATTERN
2709  014122  103003                        BCC     .+10            ;SET SIGN BIT IF CARRY=1
2710  014124  052737  100000  001340        BIS     #BIT15,PATA     ;FROM BIT 0
2711  014132  013737  001340  001342        MOV     PATA,PATB       ;SET BOTH PATTERN
2712  014140  000137  011744                JMP     CKADRS          ;CONTINUE THE VERIFY OPERATION
2713  014144  005037  001360          9$:   CLR     HEDERR          ;CLEAR HEADER CHECK ERROR INDICATOR
```

```
2714  014150  004737  015376              JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
2715  014154  004737  015710              JSR     PC,SETHDR       ;GO SET UP HEADERS IN CORE
2716  014160  013737  001322  001344      MOV     BEGCYL,CYLCK    ;USE 'BEGCYL' FOR FORMAT VERIFICATION
2717  014166  012737  177776  005614      MOV     #-2,FMTDPB+4    ;SET UP WORD COUNT
2718  014174  012737  037544  005616      MOV     #RBUF,FMTDPB+6  ;SET UP BUFFER ADRS
2719  014202  005037  005620              CLR     FMTDPB+10       ;CLEAR THE SECTOR & TRACK ADDRESS FIELD
2720  014206  013737  001322  005622      MOV     BEGCYL,FMTDPB+12  ;SETUP THE CYLINDER FIELD
2721  014214  112737  000173  005612      MOVB    #RDHD,FMTDPB+2  ;SET UP READ HEADER & DATA COMMAND
2722  014222  012737  014222  001124      MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
2723  014230  012706  001100              MOV     #$TACK,SP       ;LOAD STACK POINTER
2724  014234  004037  024736      1$:     JSR     R0,RM03         ;GO READ HEADER
2725  014240  005610                      FMTDPB                  ;ADRS OF PARAMETER TBL
2726  014242  000774                      BR      1$              ;WAIT IF QUEUE IS FULL
2727  014244  005737  005626      2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
2728  014250  001775                      BEQ     2$              ;BRANCH IF NOT DONE
2729  014252  100024                      BPL     4$              ;BR IF NOT ERROR
2730  014254  012737  014336  001174      MOV     #5$,$ESCAPE     ;ESCAPE TO 5$ ON ERROR
2731  014262  004737  015230              JSR     PC,ERINDX       ;SEE WHICH ERROR
2732  014266  104010                      ERROR   10              ;DRIVE OFFLINE
2733  014270  104011                      ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
2734  014272  104012                      ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
2735  014274  104013                      ERROR   13              ;SOFTWARE TIMEOUT
2736  014276  104014                      ERROR   14              ;DRIVE UNSAFE ERROR
2737  014300  032737  177577  005626      BIT     #^C<HCE>,FMTDPB+16  ;IS ONLY ERROR A HEADER COMPARE ERROR ?
2738  014306  001401                      BEQ     3$              ;BR IF YES
2739  014310  104021                      ERROR   21              ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
2740  014312  005037  001174      3$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
2741  014316  104022                      ERROR   22              ;HEADER COMPARE ERROR VERIFYING HEADERS
2742  014320  004737  015330              JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
2743  014324  023737  037544  037554  4$: CMP     RBUF,BUFP       ;SEE IF CYL READ EQUALS CYL EXPECTED
2744  014332  001403                      BEQ     6$              ;BR IF CYL CORRECT
2745  014334  104023                      ERROR   23              ;CYLINDER NOT CORRECT
2746  014336  004737  015330      5$:     JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
2747  014342  023737  001320  001344  6$: CMP     ENDCYL,CYLCK    ;SEE IF LAST CYLINDER
2748  014350  001002                      BNE     7$              ;BR IF NOT FINISHED
2749  014352  000137  014506              JMP     $EOP            ;END OF FORMAT
2750  014356  005237  001344      7$:     INC     CYLCK           ;INCREMENT CYLINDER ADDRESS BEING CHECKED
2751  014362  022737  001466  001344      CMP     #822.,CYLCK     ;LAST CYLINDER ?
2752  014370  003041                      BGT     8$              ;NO
2753  014372  122737  000004  005621      CMPB    #4,FMTDPB+11    ;LAST TRACK ?
2754  014400  003035                      BGT     8$              ;NO,THEN BRANCH
2755  014402  005237  001344              INC     CYLCK           ;INCREMENT CYLINDER ADDRESS
2756  014406  023737  001320  001344      CMP     ENDCYL,CYLCK    ;LAST CYLINDER ?
2757  014414  001752                      BEQ     6$              ;EXIT IF IT IS
2758  014416  005037  001344              CLR     CYLCK           ;YES LAST CYLINDER
2759  014422  005037  005622              CLR     FMTDPB+12       ;RESER CYLINDER # TO 0
2760  014426  005037  005620              CLR     FMTDPB+10       ;RESET TRACK AND SECTOR #
2761  014432  005037  005626              CLR     FMTDPB+16       ;RESET CYLINDER #
2762  014436  013746  001322              MOV     BEGCYL,-(SP)  ;SAVE BEGCYL
2763  014442  013746  001326              MOV     BEGTRK,-(SP)    ;SAVE BEGTRK
2764  014446  005037  001326              CLR     BEGTRK
2765  014452  005037  001322              CLR     BEGCYL
2766  014456  004737  015710              JSR     PC,SETHDR       ;SET UP BUFFER
2767  014462  012637  001326              MOV     (SP)+,BEGTRK    ;RESTORE BEGTRK
2768  014466  012637  001322              MOV     (SP)+,BEGCYL    ;RESTORE BEGCYL
2769  014472  000660                      BR      1$
```

```
2770  014474  004737  015776          8$:     JSR     PC,UPDACY       ;SET UP FOR NEXT CYL
2771  014500  005237  005622                  INC     FMTDPB+12       ;ADVANCE TO NEXT CYL #
2772  014504  000653                          BR      1$              ;GO READ NEXT HEADER
2773
2774                                  .SBTTL  END OF PASS ROUTINE
2775
2776                                  ;;*****************************************************************
2777                                  ;*INCREMENT THE PASS NUMBER ($PASS)
2778                                  ;*IF THERES A MONITOR GO TO IT
2779                                  ;*IF THERE ISN'T JUMP TO DONE
2780
2781  014506                          $EOP:
2782  014506  005737  001410          5$:     TST     EDIT            ;TEST LAST TRACK ?
2783  014512  001517                          BEQ     11$             ;BRANCH IF NOT
2784  014514  005037  001346                  CLR     RETRY           ;CLEAR RETRY COUNT
2785  014520  004737  016732                  JSR     PC,SORT2        ;SORT THE USTAB
2786  014524  004737  017070                  JSR     PC,TEXT         ;SET UP THE BUFFER
2787  014530  112737  000020  005611          MOVB    #20,FMTDPB+1    ;16 BIT MODE
2788  014536  112737  000143  005612          MOVB    #SETFMT,FMTDPB+2        ;SET FORMAT COMMAND
2789  014544  004037  024736          6$:     JSR     R0,RM03         ;CALL THE DRIVER
2790  014550  005610                          FMTDPB
2791  014552  000774                          BR      6$              ;LOOP IF QUEQU FAILS
2792  014554  005737  005626          7$:     TST     FMTDPB+16       ;ALL DONE ?
2793  014560  001775                          BEQ     7$              ;BRANCH IF NOT
2794  014562  012737  001466  005622          MOV     #822.,FMTDPB+12 ;CYLINDER ADDRESS
2795  014570  112737  000004  005621          MOVB    #4,FMTDPB+11    ;TRACK NUMBER
2796  014576  105037  005620                  CLRB    FMTDPB+10       ;SECTOR 0
2797  014602  012737  157700  005614          MOV     #-<32.*258.>,FMTDPB+4 ;WORD COUNT
2798  014610  012737  037554  005616          MOV     #BUFP,FMTDPB+6  ;BUFFER ADDRESS
2799  014616  112737  000163  005612          MOVB    #WRTHD,FMTDPB+2 ;WRITE HEAD AND DATA COMMAND
2800  014624  004037  024736          8$:     JSR     R0,RM03         ;CALL THE DRIVER
2801  014630  005610                          FMTDPB
2802  014632  000774                          BR      8$              ;LOOP IF QUEUE FAILS
2803  014634  005737  005626          9$:     TST     FMTDPB+16       ;ALL DONE ?
2804  014640  001775                          BEQ     9$              ;BRANCH IF NOT
2805  014642  100043                          BPL     11$             ;EXIT IF NO ERROR
2806  014644  005237  001346                  INC     RETRY           ;
2807  014650  022737  000003  001346          CMP     #3,RETRY        ;OVER 3 TIMES ?
2808  014656  101362                          BHI     8$              ;BRANCH IF IT IS
2809  014660  122737  000037  005620          CMPB    #31.,FMTDPB+10  ;LAST SECTOR ?
2810  014666  003431                          BLE     11$             ;BRANCH IF ALL SET
2811  014670  113737  005546  001350          MOVB    RM.REG+RMDA,SAVSEC      ;FOUND THE FALLING SPOT
2812  014676  001003                          BNE     10$             ;BRANCH IF NOT 0
2813  014700  012737  000037  001350          MOV     #31.,SAVSEC     ;THE LAST SECTOR IS A BAD SECTOR
2814  014706  113737  001350  005620  10$:    MOVB    SAVSEC,FMTDPB+10        ;LOAD THE NEW STARTING SECTOR
2815  014714  113701  001350                  MOVB    SAVSEC,R1               ;COOK THE WORD CTR AND BUFFER
2816  014720  006301                          ASL     R1              ;WORD INDEX
2817  014722  016137  005752  005616          MOV     ADRTBL(R1),FMTDPB+6     ;BUFFER ADDRESS
2818  014730  012737  157700  005614          MOV     #-<258.*32.>,FMTDPB+4   ;WORD COUNT
2819  014736  066137  006052  005614          ADD     WCTBL(R1),FMTDPB+4      ;WORD COUNT
2820  014744  005037  001346                  CLR     RETRY                   ;RESET THE RETRY FLAG
2821  014750  000725                          BR      8$                      ;BRANCH BACK
2822  014752  005237  001216          11$:    INC     $DEVCT          ;INCREMENT THE DEVICE COUNT
2823  014756  005737  001216                  TST     MODE            ;FORMAT OR CHECK MODE
2824  014762  001403                          BEQ     12$             ;BR IF CHECK
2825  014764  104401  033606                  TYPE    ,MFCMPT         ;FORMAT COMPLETE
```

```
2826  014770  000402                    BR      13$             ;FINISH THE MESSAGE
2827  014772  104401  033633    12$:    TYPE    ,MCCMPT         ;CHECK COMPLETE
2828  014776  104401  033657    13$:    TYPE    ,NUMERR         ;TOTAL ERRORS
2829  015002  013746  001126            MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
2830  015006  104405                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2831  015010  104401  001203            TYPE    ,$CRLF
2832  015014  005737  001404            TST     DEVMP           ;ALL ASSIGNED DEVICE DONE ?
2833  015020  001402                    BEQ     14$             ;BRANCH IF ALL DONE
2834  015022  005337  001214            DEC     $PASS           ;RESTORE THE PASS COUNT
2835  015026  000240            14$:    NOP                     ;
2836
2837
2838  015030  005237  001214            INC     $PASS           ;;INCREMENT THE PASS NUMBER
2839  015034  042737  100000  001214    BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
2840  015042  005327                    DEC     (PC)+           ;;LOOP?
2841  015044  000001            SEOPCT: .WORD   1
2842  015046  003013                    BGT     $DOAGN          ;;YES
2843  015050  012737                    MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
2844  015052  000001            SENDCT: .WORD   1
2845  015054  015044                    $EOPCT
2846  015056  013700  000042    $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
2847  015062  001405                    BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
2848  015064  000005                    RESET                   ;;CLEAR THE WORLD
2849  015066  004710            $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
2850  015070  000240                    NOP                     ;;SAVE ROOM
2851  015072  000240                    NOP                     ;;FOR
2852  015074  000240                    NOP                     ;;ACT11
2853  015076                    $DOAGN:
2854  015076  000137                    JMP     @(PC)+          ;;RETURN
2855  015100  015102            $RTNAD: .WORD   DONE
2856  015102  105737  001150    DONE:   TSTB    $AUTOB          ;RUN UNDER APT
2857  015106  001405                    BEQ     2$              ;NO
2858  015110  005737  001404            TST     DEVMP           ;ALL DRIVES ARE DONE ?
2859  015114  001427                    BEQ     4$              ;BRANCH IF ALL DONE
2860  015116  000137  007360            JMP     MSFX            ;NEXT DRIVE FROM APT DEVICE MAP
2861  015122  032777  000001  164024    BIT     #SW0,@SWR       ;SEE IF SWR 0 SET
2862  015130  001002                    BNE     1$              ;BR IF SET
2863  015132  000137  010106            JMP     M0              ;ASK FOR NEXT DRIVE
2864  015136  012706  001100    1$:     MOV     #STACK,SP       ;RESET STACK POINTER
2865  015142  005737  001316            TST     MODE            ;VERIFY MODE ?
2866  015146  001402                    BEQ     3$              ;YES
2867  015150  000137  011744            JMP     CKADRS          ;DO THE FORMAT OR CHECK AGAIN
2868  015154  012737  133331  001340    MOV     #133331,PATA    ;INITIATE PATTERN
2869  015162  012737  133331  001342    MOV     #133331,PATB    ;INITIATE PATTERN
2870  015170  000137  011744            JMP     CKADRS          ;LOOP ON CHECK
2871  015174  005237  001214    4$:     INC     $PASS           ;GARBAGE CODE FOR APT
2872  015200  005237  001212    5$:     INC     $TESTN
2873  015204  005237  001216            INC     $DEVCT
2874  015210  000773                    BR      5$
2875  015212  104401  001203    REJEC1: TYPE    ,$CRLF
2876  015216  104401  034365            TYPE    ,MESG4
2877  015222  000000                    HALT
2878  015224  000137  000200            JMP     @#200
2879
2880                    ;;**************************************************************
2881
```

```
                                .SBTTL  SUPPORT SUBROUTINES

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

                                ;THIS ROUTINE DETERMINES THE ERROR TYPE

015230  010146          ERINDX: MOV     R1,-(SP)        ;STORE R1
015232  005001                  CLR     R1              ;CLEAR R1
015234  033727  005626          BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
015240  060006                  .WORD   BIT14!BIT13!BIT2!BIT1  ;DRIVE OFFLINE
015242  001025                  BNE     5$              ;BR IF OFFLINE
015244  033727  005626          BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
015250  010000                  .WORD   BIT12           ;DRIVE PERSISTENTLY UNSAFE
015252  001020                  BNE     4$              ;BR IF UNSAFE
015254  033727  005626          BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
015260  006000                  .WORD   BIT11!BIT10     ;UNCORRECTABLE MASSBUS PARITY ERROR ?
015262  001013                  BNE     3$              ;BR IF PARITY ERROR
015264  033727  005626          BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
015270  001400                  .WORD   BIT09!BIT08     ;SOFTWARE TIMEOUT ?
015272  001006                  BNE     2$              ;BR IF YES
015274  033727  005626          BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
015300  000020                  .WORD   BIT04           ;DRIVE UNSAFE ERROR ?
015302  001001                  BNE     1$              ;BR IF YES
015304  005201                  INC     R1              ;INCREMENT THE RETURN INDEX
015306  005201          1$:     INC     R1              ;INCREMENT THE RETURN INDEX
015310  005201          2$:     INC     R1              ;INCREMENT THE RETURN INDEX
015312  005201          3$:     INC     R1              ;INCREMENT THE RETURN INDEX
015314  005201          4$:     INC     R1              ;INCREMENT THE RETURN INDEX
015316  006301          5$:     ASL     R1              ;DOUBLE THE INCREMENT
015320  060166  000002          ADD     R1,2(SP)        ;DEVELOP THE RETURN ADDRESS
015324  012601                  MOV     (SP)+,R1        ;RESTORE R1
015326  000207                  RTS     PC              ;RETURN

                                ;ROUTINE TO CHECK FOR LOOP ON ERROR

015330  032777  001000  163616  LOP.CK: BIT     #SW09,@SWR      ;LOOP ON ERROR ?
015336  001402                  BEQ     1$              ;BR IF NOT
015340  000177  163560          JMP     @SLPERR         ;GO TO THE LOOP ON ERROR ADDRESS
015344  005037  001174  1$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
015350  033727  005626          BIT     FMTDPB+16,(PC)+ ;CHECK FOR 'FATAL' ERROR
015354  072002                  .WORD   BIT14!BIT13!BIT12!BIT10!BIT01  ;'FATAL' ERROR BITS
015356  001004                  BNE     2$              ;BR IF NOT
015360  032737  004000  005554  BIT     #WLE,RM.REG+RMER!  ;WRITE LOCK ERROR ?
015366  001402                  BEQ     3$              ;BR IF NOT
015370  000137  014506  2$:     JMP     $EOP            ;TERMINATE THE FORMAT
015374  000207          3$:     RTS     PC              ;RETURN

                                ;THIS ROUTINE SETS UP THE INPUT TABLE FOR THE DRIVER ROUTINE

015376  113737  001220  005610  SETTBL: MOVB    DRIVE,FMTDPB    ;SET UP DRIVE #
015404  105037  005613          CLRB    FMTDPB+3        ;CLEAR HIGH ORDER ADRS BITS
015410  013737  001356  005614  MOV     MWC,FMTDPB+4    ;LOAD UP WORD COUNT
015416  012737  037554  005616  MOV     #BUFP,FMTDPB+6  ;LOAD UP CURRENT ADRS
015424  105037  005620          CLRB    FMTDPB+10       ;SET SECTOR TO ZERO
015430  113737  001326  005621  MOVB    BEGTRK,FMTDPB+11  ;SET UP STARTING TRK ADRS
015436  013737  001322  005622  MOV     BEGCYL,FMTDPB+12  ;SET UP STARTING CYL
```

```
2938  015444  005037  005626                  CLR     FMTDPB+16       ;CLEAR RM03 STATUS
2939  015450  013737  001326  001334          MOV     BEGTRK,TRKCNT   ;SET UP PARTIAL CYL TRACK COUNT
2940  015456  013737  001330  001332          MOV     TTRKS,TTRKSC    ;SET UP TOTAL TRACKS COUNTER
2941  015464  000207                          RTS     PC              ;RETURN FROM SETUP
2942
2943                                  ;THIS ROUTINE CONTROLS THE DISK ADDRESSING AND TOTAL TRK COUNT
2944                                  ;IT IS ENTERED AFTER EVERY TRK OPERATION
2945
2946  015466  005337  001332          TRKTST: DEC     TTRKSC          ;SEE IF LAST TRACK
2947  015472  001472                          BEQ     3$              ;BRANCH IF LAST TRACK
2948  015474  022737  000004  001334          CMP     #4.,TRKCNT      ;IS THIS THE LAST TRACK IN THE CYLINDER ?
2949  015502  001423                          BEQ     1$              ;BRANCH IF SO
2950  015504  005237  001334                  INC     TRKCNT          ;COUNT UP TRACK WITHIN CYLINDER
2951  015510  105237  005621                  INCB    FMTDPB+11       ;INCREMENT TRACK NUMBER
2952  015514  022737  001466  005622          CMP     #822.,FMTDPB+12 ;LAST CYLINDER AND LAST TRACK ?
2953  015522  003010                          BGT     5$              ;NO
2954  015524  122737  000004  005621          CMPB    #4,FMTDPB+11    ;LAST TRACK ?
2955  015532  003004                          BGT     5$              ;NO
2956  015534  005337  001332                  DEC     TTRKSC          ;DECREMENT ONE TRACK
2957  015540  001447                          BEQ     3$              ;BRANCH IF LAST TRACK
2958  015542  000403                          BR      1$              ;DON'T DESTORY THE LAST TRACK OF CYL822
2959  015544  004737  016054          5$:     JSR     PC,UPDATK       ;UPDATE TRACK ADDRESS IN BUFFER
2960  015550  000441                          BR      2$              ;EXIT
2961  015552  005037  001334          1$:     CLR     TRKCNT          ;CLEAR TRACK COUNT FOR NEXT CYLINDER
2962  015556  105037  005621                  CLRB    FMTDPB+11       ;RESET TRACK ADDRESS TO 0
2963  015562  005237  005622                  INC     FMTDPB+12       ;UPDATE CYLINDER NUMBER
2964  015566  022737  001467  005622          CMP     #823.,FMTDPB+12 ;LAST CYLINDER REACHED ?
2965  015574  003025                          BGT     4$              ;NO
2966  015576  013746  001322                  MOV     BEGCYL,-(SP)    ;SAVE BEGCYL
2967  015602  013746  001326                  MOV     BEGTRK,-(SP)    ;SAVE BEGTRK
2968  015606  005037  001322                  CLR     BEGCYL          ;RESET THE STARTING CYL
2969  015612  005037  001326                  CLR     BEGTRK          ;RESET THE START TRACK
2970  015616  004737  015710                  JSR     PC,SETHDR       ;INITIATE HEAD WORDS
2971  015622  005037  005620                  CLR     FMTDPB+10       ;CLEAR TRACK AND SECTOR
2972  015626  005037  005622                  CLR     FMTDPB+12       ;CLEAR CYLINDER #
2973  015632  005037  005626                  CLR     FMTDPB+16       ;CLEAR STATUS
2974  015636  012637  001326                  MOV     (SP)+,BEGTRK    ;RESTORE BEGTRK
2975  015642  012637  001322                  MOV     (SP)+,BEGCYL    ;RESTORE BEGCYL
2976  015646  000402                          BR      2$
2977  015650  004737  015776          4$:     JSR     PC,UPDACY       ;UPDATE CYLINDER NUMBER IN BUFFER
2978  015654  062716  000002          2$:     ADD     #2,(SP)         ;ADD TWO TO RETURN ADRS
2979  015660  000207                  3$:     RTS     PC              ;RETURN
2980
2981                                  ;THIS ROUTINE SETS UP WC & BA FOR REMAINING SECTORS
2982                                  ;AFTER A WRITE CHECK ERROR
2983
2984  015662  013737  001352  005614  SCAWC:  MOV     SAVWC,FMTDPB+4  ;SET UP WC FOR REMAINING SECTORS
2985  015670  062737  001004  005616          ADD     #516.,FMTDPB+6  ;SET UP BA FOR REMAINING SECTORS
2986  015676  005237  005620                  INC     FMTDPB+10       ;ADVANCE TO NEXT SECTOR IN TBL
2987  015702  005037  001314                  CLR     SOFSW           ;RESET RETRY COUNTER
2988  015706  000207                          RTS     PC              ;RETURN TO COMPLETE TRK FORMAT
2989
2990
2991                                  ;THIS ROUTINE SETS UP THE CYLINDER ADRS, FORMAT BIT, TRACK AND
2992                                  ;SECTOR ADRS IN MEMORY WITH THE STARTING CYL - TRK INFORMATION
2993
```

```
2994  015710  013701  001364          SETHDR: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
2995  015714  012737  000001  001212          MOV     #1,$TESTN       ;GARBAGE CODE FOR APT
2996  015722  012700  037554                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
2997  015726  013702  001322                  MOV     BEGCYL,R2       ;PUT STARTING CYL # IN R2
2998  015732  052702  140000                  BIS     #140000,R2      ;BIT 15,BIT 14 ARE SET FOR 144 SPC 3/14/77
2999  015736  013703  001326                  MOV     BEGTRK,R3       ;PUT STARTING TRK # IN R3
3000  015742  000303                          SWAB    R3              ;JUSTIFY TRACK ADRS
3001  015744  005737  001362                  TST     SEC30           ;SEE IF 30 OR 32 SECTOR MODE
3002  015750  001402                          BEQ     1$              ;BR IF 30 SECTOR MODE
3003  015752  052702  010000                  BIS     #10000,R2       ;SET THE 32 SECTOR FORMAT BIT
3004  015756  010220                  1$:     MOV     R2,(R0)+        ;WRITE IN HEADER AREA OF CORE THE CYL ADRS
3005  015760  010320                          MOV     R3,(R0)+        ;WRITE IN HEADER AREA OF CORE THE TRK ADRS
3006  015762  062700  001000                  ADD     #512.,R0        ;SET UP FOR NEXT HEADER
3007  015766  005203                          INC     R3              ;UPDATE SECTOR ADRS FOR NEXT HEADER
3008  015770  005301                          DEC     R1              ;MAINTAIN COUNT OF SECTORS
3009  015772  002371                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
3010  015774  000207                          RTS     PC              ;EXIT - HEADERS ARE LOADED INTO CORE
3011
3012                          ;THIS ROUTINE UPDATES THE CYLINDER ADRS OF THE HEADER WORDS IN CORE
3013
3014  015776  013701  001364          UPDACY: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
3015  016002  012700  037554                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
3016  016006  010246                          MOV     R2,-(SP)        ;SAVE R2
3017  016010  005220                  1$:     INC     (R0)+           ;INCREMENT FOR NEXT CYLINDER
3018  016012  042720  177400                  BIC     #177400,(R0)+   ;RESET TRK ADRS TO 0
3019  016016  012702  001000                  MOV     #512.,R2        ;DATA FIELD LENGTH
3020  016022  013720  001340          2$:     MOV     PATA,(R0)+      ;FILL BUFFER
3021  016026  013720  001342                  MOV     PATB,(R0)+      ;FILL BUFFER
3022  016032  162702  000004                  SUB     #4,R2           ;END OF DATA FIELD ?
3023  016036  001371                          BNE     2$              ;NO
3024  016040  005301                          DEC     R1              ;COUNT SECTORS
3025  016042  002362                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
3026  016044  012602                          MOV     (SP)+,R2        ;RESETORE THE R2
3027  016046  005237  001212                  INC     $TESTN          ;INCREMENT TEST NUMBER FOR EACH CYLINDER CHANGE
3028                                                                  ;GARBADE CODE FOR APT
3029  016052  000207                          RTS     PC              ;EXIT
3030
3031                          ;THIS ROUTINE UPDATES THE TRACK ADRS OF THE HEADER WORDS IN CORE
3032
3033  016054  013701  001364          UPDATK: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
3034  016060  012700  037554                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
3035  016064  005720                          TST     (R0)+           ;POINT HEADER POINTER TO TRK - SEC ADRS
3036  016066  062710  000400          1$:     ADD     #400,(R0)       ;INDEX TRK ADRS
3037  016072  062700  001004                  ADD     #516.,R0        ;SET UP FOR NEXT HEADER
3038  016076  005301                          DEC     R1              ;COUNT SECTORS
3039  016100  002372                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
3040  016102  000207                          RTS     PC              ;EXIT
3041
3042                          ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
3043
3044  016104  012737  016162  000004  ST.CLK: MOV     #STCLK1,@#ERRVEC  ;SET UP VECTOR FOR P CLK
3045  016112  005037  000006                  CLR     @#ERRVEC+2      ;NEW PSW
3046  016116  005777  163154                  TST     @#LKCSR         ;CHECK FOR KW11-P
3047  016122  013746  001302                  MOV     $LPVEC,-(SP)    ;VECTOR ADDRESS
3048  016126  012776  016312  000000          MOV     #CLOCK,@(SP)    ;SET UP KW11-P VECTOR
3049  016134  062716  000002                  ADD     #2,(SP)         ;POINT TO PSW
```

```
3050  016140  012736  000300          MOV     #PR6,@(SP)+     ;PSW - PRI 6
3051  016144  012777  177777  163126  MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER
3052  016152  012777  000135  163116  MOV     #135,@$LKCSR    ;SET CLK - CNT UP
3053  016160  000426                  BR      STCLK3
3054  016162  062706  000004  STCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
3055  016166  012737  016232  000004  MOV     #STCLK2,@#ERRVEC        ;CHANGE ERROR VECTOR
3056  016174  005777  163106          TST     @$LKS           ;LOOK FOR KW11-L
3057  016200  013746  001310          MOV     $LLVEC,-(SP)    ;KW11-L VECTOR ADDRESS
3058  016204  012776  016312  000000  MOV     #CLOCK,@(SP)    ;SET UP KW11-L VECTOR
3059  016212  062716  000002          ADD     #2,(SP)         ;INCREMENT VECTOR ADDRESS
3060  016216  012736  000300          MOV     #PR6,@(SP)+     ;PSW - PRI 6
3061  016222  012777  000100  163056  MOV     #100,@$LKS      ;SET KW11-L INTERRUPT ENABLE
3062  016230  000402                  BR      STCLK3
3063  016232  062706  000004  STCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
3064  016236  012737  000006  000004  STCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
3065  016244  000207                  RTS     PC
3066
3067                          ;THIS CODE PRINT THE ABORT MESSAGE AND LET O.P. RESTART
3068                          ;
3069  016246  105737  001150  RESTRT: TSTB    $AUTOB          ;RUN UNDER APT ?
3070  016252  001016                  BNE     1$              ;BRANCH IF SO
3071  016254  004737  021772          JSR     PC,$TKINT       ;ENABLE TO READ
3072  016260  104401  034204          TYPE    ,MESG2          ;IMPORPER MFG INFORMATION
3073  016264  104411                  RDLIN
3074  016266  012601                  MOV     (SP)+,R1        ;LOCATE THE READ IN LINE
3075  016270  105711                  TSTB    (R1)            ;TERMINATOR BY <CR>
3076  016272  001406                  BEQ     1$              ;BRANCH IF SO
3077  016274  121127  000131          CMPB    (R1),#'Y        ;ENTER Y ?
3078  016300  001403                  BEQ     1$              ;BRANCH IF SO
3079  016302  000000                  HALT                    ;OTHERWISE HALT
3080  016304  000137  000200          JMP     @#200           ;JUMP TO 200 IF HIT START SWITCH
3081  016310  000207          1$:     RTS     PC              ;EXIT
3082
3083                          ;THIS CODE SERVICES A CLOCK INTERRUPT EVERY 16MS
3084
3085  016312  012746  000020  CLOCK:  MOV     #16.,-(SP)      ;PUT MILLISECONDS ON THE STACK
3086  016316  004737  030404          JSR     PC,$RPTMR       ;GO REPORT TIME
3087  016322  000002                  RTI                     ;RETURN AND CONTINUE
3088
3089                          ;ROUTINE TO INTERCEPT 'CONTROL C' TYPED DURING PARAMETER ENTRY TIME
3090
3091  016324  012706  001100  OENTER: MOV     #STACK,SP       ;INITIALIZE THE STACK
3092  016330  005737  024126  1$:     TST     TRNSWT          ;ALL ACTIVITY STOPPED ?
3093  016334  001375                  BNE     1$              ;BR IF NOT
3094  016336  000005                  RESET                   ;CLEAR THE BUS
3095  016340  000137  010046          JMP     M1B             ;START AGAIN WITH DRIVE SELECTION
3096
3097                          ;ROUTINE TO TYPE THE PRESENT DISK ADDRESS.  ENTERED BY TYPING 'CONTROL C'
3098
3099  016344  117746  162612  NEWSVC: MOVB    @$TKB,-(SP) ;READ FROM TTY BUFFER
3100  016350  042716  177600          BIC     #^C177,(SP)     ;STRIP PARITY
3101  016354  022627  000017          CMP     (SP)+,#15.      ;CONTROL-O ?
3102  016360  001406                  BEQ     1$              ;YES
3103  016362  005777  162574          TST     @$TKB           ;CLEAR DONE BIT
3104  016366  012777  000100  162564  MOV     #100,@$TKS      ;ENABLE TTY INTERRUPT
3105  016374  000002                  RTI
```

```
 3106  016376  004737  021772          1$:     JSR     PC,$TKINT       ;INITIAL VECTOR
 3107  016402  005037  177776          TYPADR: CLR     PSW             ;SET PROCESSOR TO PRIORITY 0
 3108  016406  012737  016324  001376          MOV     #0ENTER,CNTLC   ;CHANGE 'CONTROL C' RETURN ADDRESS
 3109  016414  104401  033707                  TYPE    ,ADDRIS         ;'PRESENT ADDRESS IS: '
 3110  016420  104401  032473                  TYPE    ,C              ;'C'
 3111  016424  013746  005622                  MOV     FMTDPB+12,-(SP) ;PUT THE CYLINDER ADDRESS ON THE STACK
 3112  016430  004737  023436                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
 3113  016434  004737  023376                  JSR     PC,$SUPRS       ;TYPE IT
 3114  016440  104401  032503                  TYPE    ,LINSP          ;SPACES
 3115  016444  104401  032475                  TYPE    ,T              ;'T'
 3116  016450  005046                          CLR     -(SP)           ;CLEAR THE STACK
 3117  016452  113716  005621                  MOVB    FMTDPB+11,(SP)  ;PUT THE TRACK ADDRESS ON THE STACK
 3118  016456  004737  023436                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
 3119  016462  004737  023376                  JSR     PC,$SUPRS       ;TYPE IT
 3120  016466  104401  001203                  TYPE    ,$CRLF          ;CR-LF
 3121  016472  012737  016344  000060          MOV     #NEWSVC,@#TKVEC ;RESTORE ENTRANCE TO THIS ROUTINE
 3122  016500  000002                          RTI                     ;RETURN
 3123
 3124                                  ;PARAMETER ENTRY ROUTINE
 3125                                  ;CALL
 3126                                  ;       MOV     #ADR,R3         ;PARAMETER TABLE ADDRESS
 3127                                  ;       JSR     PC,PARENT       ;GET THE PARAMETERS
 3128
 3129  016502  010346                  PARENT: MOV     R3,-(SP)        ;SAVE R3
 3130  016504  012703  005650                  MOV     #TABLE,R3       ;PARAMETER TABLE ADDRESS
 3131  016510  012337  016520          1$:     MOV     (R3)+,3$        ;ADDRESS OF PARAMETER NAME
 3132  016514  001436                          BEQ     9$              ;BR IF AT END OF TABLE
 3133  016516  104401                          TYPE                    ;TYPE THE PARAMETER NAME
 3134  016520  000000                  3$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
 3135  016522  012302                          MOV     (R3)+,R2        ;MAXIMUM PARAMETER VALUE
 3136  016524  012305                          MOV     (R3)+,R5        ;ADDRESS OF PARAMETER
 3137  016526  011546                          MOV     (R5),-(SP)      ;CURRENT VALUE OF PARAMETER
 3138  016530  104405                          TYPDS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
 3139  016532  104401  032467                  TYPE    ,SLASH          ;' /'
 3140  016536  104411                          RDLIN                   ;READ THE KEYBOARD
 3141  016540  012601                          MOV     (SP)+,R1        ;INPUT ASCII STRING ADDRESS
 3142  016542  004537  017730                  JSR     R5,CK.DIG       ;CHECK THE DIGIT(S)
 3143  016546  016510                          1$                      ;CARRIAGE RETURN ONLY ENTERED
 3144  016550  016612                          9$                      ;PERIOD ONLY ENTERED
 3145  016552  016566                          6$                      ;ILLEGAL INPUT
 3146  016554  016562                          5$                      ;TERMINATED WITH A CARRIAGE RETURN
 3147  016556  016566                          6$                      ;TERMINATED WITH A "."
 3148  016560  016600                          7$                      ;TERMINATED WITH A "-"
 3149  016562  010215                  5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
 3150  016564  000751                          BR      1$              ;GET MORE PARAMETERS
 3151  016566  104401  033230          6$:     TYPE    ,BADENT         ;'BAD ENTRY'
 3152  016572  162703  000006                  SUB     #6,R3           ;DECREMENT THE TABLE POINTER
 3153  016576  000744                          BR      1$              ;TRY AGAIN
 3154  016600  010215                  7$:     MOV     R2,(R5)         ;NEW VALUE
 3155  016602  000403                          BR      9$              ;EXIT
 3156  016604  012703  005650          8$:     MOV     #TABLE,R3       ;RELOAD THE PARAMETER TABLE ADDRESS
 3157  016610  000737                          BR      1$              ;TRY AGAIN
 3158  016612  012603                  9$:     MOV     (SP)+,R3        ;RESTORE R3
 3159  016614  000207                          RTS     PC              ;RETURN
 3160                                  ;THIS ROUTINE LOAD THE BAD SECTOR INTO USTAB TABLE
 3161                                  ;CALL SEQ  JSR  PC,RECORD
```

```
                                        ;ALL REGISTER USED ARE DESTORIED
016616  012701  003434          RECORD: MOV     #USTAB+10,R1        ;TABLE ENTRY IN R1
016622  022711  177777          1$:     CMP     #-1,(R1)            ;AN OPENING IN THE TABLE
016626  001421                          BEQ     3$                  ;BRANCH IF IT IS
016630  023711  005622                  CMP     FMTDPB+12,(R1) ;CYLINDER NUMBER MATCHES ?
016634  001010                          BNE     2$                  ;BRANCH IF NOT
016636  123761  005621  000003          CMPB    FMTDPB+11,3(R1) ;TRK NUMBER MATCHES ?
016644  001004                          BNE     2$                  ;BRANCH IF NOT
016646  123761  005620  000002          CMPB    FMTDPB+10,2(R1) ;SECTOR NUMBER MATCHES ?
016654  001416                          BEQ     4$                  ;BRANCH TO EXIT,IF THE SPOT HAS RECOREDED
016656  062701  000004          2$:     ADD     #4,R1               ;INCREMENT 4 BYTES
016662  022701  004424                  CMP     #USTAB+512.,R1      ;END OF TABLE ?
016666  101355                          BHI     1$                  ;BRANCH IF NOT
016670  000411                          BR      5$                  ;THROW AWAY THE PACK
016672  013711  005622          3$:     MOV     FMTDPB+12,(R1)      ;LOAD THE CYLINDER #
016676  113761  005621  000003          MOVB    FMTDPB+11,3(R1)     ;LOAD THE TRK NUMBER
016704  113761  001350  000002          MOVB    SAVSEC,2(R1)        ;LOAD THE SECTOR NUMBER
016712  000207                  4$:     RTS     PC                  ;EXIT
016714  104401  001203          5$:     TYPE    ,$CRLF
016720  104401  034103                  TYPE    ,MESG3              ;THROW AWAY THE PACK
016724  000000                          HALT
016726  000137  000200                  JMP     @#200               ;RESTART IF DESIRED ?

                                        ;THIS ROUTINE SORT THE USTAB IN ASCENDING ORDER
                                        ;CALL SEQ       JSR     PC,SORT2

016732  012701  003434          SORT2:  MOV     #USTAB+8.,R1        ;TOP POINTER
016736  012702  000175                  MOV     #125.,R2            ;TOTAL 126 ELEMENTS
016742  012703  004420                  MOV     #USTAB+508.,R3      ;THE LAST ELEMENT
016746  022711  177777                  CMP     #-1,(R1)            ;THE TABLE IS EMPTY ?
016752  001445                          BEQ     4$                  ;QUICK EXIT
016754  021161  000004          1$:     CMP     (R1),4(R1)          ;COMPARE THE CYLINDER NUMBER
016760  101013                          BHI     2$                  ;SWITCH THE TWO ELEMENT
016762  103426                          BLO     3$                  ;INCREMENT POINTER
016764  126161  000003  000007          CMPB    3(R1),7(R1)         ;COPMARE THE TRK NUMBER
016772  101006                          BHI     2$                  ;SWITCH ELEMENT
016774  103421                          BLO     3$                  ;INCREMENT POINTER
016776  126161  000002  000006          CMPB    2(R1),6(R1)         ;COMPARE THE SECTOR NUMBER
017004  101001                          BHI     2$                  ;SWITCH ELEMENT
017006  000414                          BR      3$                  ;INCREMENT POINTER
017010  011146                  2$:     MOV     (R1),-(SP)          ;SAVE THE N TH BLOCK
017012  016146  000002                  MOV     2(R1),-(SP)         ;INTO STACK
017016  016111  000004                  MOV     4(R1),(R1)          ;SWITCH THE N+1 BLOCK TO N BLOCK
017022  016161  000006  000002          MOV     6(R1),2(R1)         ;
017030  012661  000006                  MOV     (SP)+,6(R1)         ;LOAD THE N+1 BLOCK
017034  012661  000004                  MOV     (SP)+,4(R1)         ;
017040  062701  000004          3$:     ADD     #4,R1               ;UPDATE THE TOP POINTER
017044  020103                          CMP     R1,R3               ;REACH THE BOTTOM ?
017046  001342                          BNE     1$                  ;BRANCH IF NOT
017050  005302                          DEC     R2                  ;DECREMENT ONE SORT COUNT
017052  001405                          BEQ     4$                  ;BRANCH IF ALL DONE
017054  162703  000004                  SUB     #4,R3               ;ASJUST THE BOTTOM POINTER
017060  012701  003434                  MOV     #USTAB+8.,R1        ;RESET TOP POINTER
017064  000733                          BR      1$                  ;BRANCH IF NOT ALL DONE
017066  000207                  4$:     RTS     PC                  ;EXIT
```

```
                                        ;THIS ROUTINE TEXT THE LAST TRACK   CYL 822, TRK 4
                                        ;SECTORS 0,2,4,6,8       16 BIT MFG
                                        ;SECTORS 1,3,5,7,9       18 BIT MFG
                                        ;SECTORS 10,12,14,16,.....30,  16 BIT USER
                                        ;SECTORS 11,13,15,17,.....31         18 BIT USER

017070  012700  037554        TEXT:   MOV     #BUFP,R0        ;BUFFER ADDRESS
017074  012701  020100                MOV     #<258.*32.>,R1  ;TOTAL WORD COUNT
017100  012702  177777                MOV     #-1,R2          ;SET ALL LOCATIONS TO -1
017104  010220                1$:     MOV     R2,(R0)+        ;FULL THE BUFFER
017106  005301                        DEC     R1              ;ALL DONE ?
017110  001375                        BNE     1$              ;LOOP, IF NOT
017112  013746  001212                MOV     $TESTN,-(SP)    ;SAVE TESTN,BEGCYL,BEGTRK
017116  013746  001322                MOV     BEGCYL,-(SP)    ;
017122  013746  001326                MOV     BEGTRK,-(SP)    ;
017126  013746  001364                MOV     MAXSEC,-(SP)    ;SAVE FLAG MAXSEC AND SEC30
017132  013746  001362                MOV     SEC30,-(SP)     ;
017136  012737  000037  001364        MOV     #37,MAXSEC      ;SET MAX 31 SECTORS
017144  012737  177777  001362        MOV     #-1,SEC30       ;ALWAYS IN 16 BIT MODE
017152  012737  001466  001322        MOV     #822.,BEGCYL    ;LOAD LAST CYLINDER
017160  012737  000004  001326        MOV     #4,BEGTRK       ;LAST TRACK
017166  004737  015710                JSR     PC,SETHDR       ;SET UP THE HEAD FIELD IN THE BUFFER
017172  012637  001362                MOV     (SP)+,SEC30     ;RESTORE SEC30 AND MAXSEC
017176  012637  001364                MOV     (SP)+,MAXSEC    ;
017202  012637  001326                MOV     (SP)+,BEGTRK    ;
017206  012637  001322                MOV     (SP)+,BEGCYL    ;
017212  012637  001212                MOV     (SP)+,$TESTN    ;
017216  012701  001414                MOV     #BAD16,R1       ;LOAD SERIAL NUMBER TO EVERY SECTOR
017222  012702  037560                MOV     #BUFP+4,R2      ;BUFFER LOCATION + 4
017226  012703  000040                MOV     #32.,R3         ;TOTAL 32 SECTORS
017232  011112                2$:     MOV     (R1),(R2)       ;1 ST SN #
017234  016162  000002  000002        MOV     2(R1),2(R2)     ;2 ND SN #
017242  005062  000004                CLR     4(R2)           ;THIRD WORD = 0
017246  005062  000006                CLR     6(R2)           ;ID = DATA PACK
017252  062702  001004                ADD     #516.,R2        ;ADVANCE TO NEXT SECTOR
017256  005303                        DEC     R3              ;DECREMENT ONE SECTOR COUNT
017260  001364                        BNE     2$              ;BRANCH IF NOT DONE
017262  005046                        CLR     -(SP)           ;LOAD TABLE INTO SECTORS
017264  005046                        CLR     -(SP)                           ;DUMMY PAIRS
017266  005046                        CLR     -(SP)
017270  012746  001414                MOV     #BAD16,-(SP)    ;TABLE ADDRESS
017274  012746  037560                MOV     #BUFP+4,-(SP)   ;BUFFER ADDRESS
017300  012746  000005                MOV     #5,-(SP)        ;SECTOR COUNT
017304  012746  002420                MOV     #BAD18,-(SP)    ;TABLE ADDRESS
017310  012746  040564                MOV     #BUFP+4+516.,-(SP) ;BUFFER ADDRESS
017314  012746  000005                MOV     #5,-(SP)        ;SECTOR COUNT
017320  012746  003424                MOV     #USTAB,-(SP)    ;USER DEFINED TABLE
017324  012746  051630                MOV     #BUFP+4+<516.*10.>,-(SP)        ;BUFFER ADDRESS
017330  005737  001362                TST     SEC30
017334  100402                        BMI     3$              ;BRANCH IF IN 16 BIT MODE
017336  012716  052634                MOV     #BUFP+4+<516.*11.>,(SP) ;RESET THE BUFFER ADDRESS
017342  012746  000013        3$:     MOV     #11.,-(SP)      ;SECTOR COUNT
017346  012701  004430                MOV     #USSAV,R1       ;TABLE ADDRESS IN R1
017352  012702  052634                MOV     #BUFP+4+<516.*11.>,R2   ;BUFFER ADDRESS IN R2
017356  005737  001362                TST     SEC30           ;BRANCH IF IN 16 BIT MODE
017362  100402                        BMI     4$              ;
```

```
3274  017364  012702  051630                MOV     #BUFP+4+<516.*10.>,R2   ;BUFFER ADDRESS FOR 18 BIT MODE
3275  017370  012703  000013        4$:     MOV     #11.,R3                 ;SECTOR COUNT IN R3
3276  017374  010105                        MOV     R1,R5                   ;TEMPOR STORAGE
3277  017376  012704  000400        5$:     MOV     #256.,R4                ;WORD COUNT = DATA FIELD OF ONE SECTOR
3278  017402  012122                6$:     MOV     (R1)+,(R2)+             ;LOAD TABLE INTO SECTOR DATA FIELD
3279  017404  005304                        DEC     R4                      ;ALL DONE ?
3280  017406  001375                        BNE     6$                      ;BRANCH IF NOT
3281  017410  010501                        MOV     R5,R1                   ;RELOAD THE TABLE ADDRESS
3282  017412  062702  001010                ADD     #520.,R2                ;SKIP ONE SECTOR
3283  017416  005303                        DEC     R3                      ;DECREMENT ONE SECTOR COUNT
3284  017420  001366                        BNE     5$                      ;BRANCH IF NOT DONE
3285  017422  012603                        MOV     (SP)+,R3                ;RETRIEVE NEXT SET OF SECTOR #
3286  017424  012602                        MOV     (SP)+,R2                ;BUFFER ADDRESS
3287  017426  012601                        MOV     (SP)+,R1                ;TABLE ADDRESS
3288  017430  010105                        MOV     R1,R5                   ;TEMPOR STORAGE
3289  017432  001361                        BNE     5$                      ;BRANCH IF NOT DUMMY PAIR
3290  017434  000207                        RTS     PC                      ;EXIT
3291
3292                                ;THIS ROUTINE RESET:
3293                                ;RESET THE BIT 14 OF THE 1 ST HEADER WORD OF THE BAD SPOT
3294                                ;CALLING SEQ;           JSR     PC,RESET
3295                                ;
3296
3297  017436  013737  005622  037544  RESET:  MOV     FMTDPB+12,RBUF  ;CYLINDER ADDRESS
3298  017444  113737  005621  037547          MOVB    FMTDPB+11,RBUF+3        ;TRACK ADDRESS
3299  017452  113737  001350  037546          MOVB    SAVSEC,RBUF+2   ;SECTOR ADDRESS
3300  017460  053737  001412  037544          BIS     HEADX,RBUF      ;SET MFG BIT,RESET USER BIT
3301  017466  113737  001220  005630          MOVB    DRIVE,FMTX      ;SETUP THE DPB FOR OPERATION
3302  017474  012737  177776  005634          MOV     #-2,FMTX+4      ;WORD COUNT =2
3303  017502  112737  000163  005632          MOVB    #WRTHD,FMTX+2   ;WRITE HEAD AND DATA COMMAND
3304  017510  012737  037544  005636          MOV     #RBUF,FMTX+6    ;BUFFER ADDRESS
3305  017516  113737  001350  005640          MOVB    SAVSEC,FMTX+10  ;SECTOR ADDRESS
3306  017524  113737  005621  005641          MOVB    FMTDPB+11,FMTX+11 ;TRACK ADDRESS
3307  017532  013737  005622  005642          MOV     FMTDPB+12,FMTX+12 ;CYLINDER ADDRESS
3308  017540  052737  010000  037544          BIS     #10000,RBUF     ;ASSUM 16 BIT MODE
3309  017546  005737  001362                  TST     SEC30           ;IN 16 BIT MODE ?
3310  017552  100403                          BMI     1$              ;BRANCH IF IT IS
3311  017554  042737  010000  037544          BIC     #10000,RBUF     ;RESET THE FMT BIT
3312  017562  004037  024736          1$:     JSR     R0,RM03         ;CALL THE DRIVER
3313  017566  005630                          FMTX                    ;DPB ADDRESS
3314  017570  000774                          BR      1$              ;BRANCH IF QUEUE FAILS
3315  017572  005737  005646          2$:     TST     FMTX+16         ; DONE ?
3316  017576  001775                          BEQ     2$              ;BRANCH IF NOT
3317  017600  000207                          RTS     PC              ;EXIT
3318
3319                                ;*************************
3320
3321                                ;THIS ROUTINE IS USED TO CHECK IF AN
3322                                ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
3323                                ;CALL
3324                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3325                                ;       JSR     R5,CK.OCT       ;CHECK THE CHARACTER
3326                                ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
3327                                ;       RETURN2                 ;CHARACTER IS IN R2 AS A
3328                                ;                               ;OCTAL DIGIT
3329
```

```
3330  017602  121127  000060  CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
3331  017606  103407                  BLO     1$              ;YES -- BRANCH
3332  017610  121127  000067          CMPB    (R1),#'7        ;GREATER THAN SEVEN?
3333  017614  101004                  BHI     1$              ;YES -- BRANCH
3334  017616  111102                  MOVB    (R1),R2         ;GET THE CHARACTER
3335  017620  042702  177770          BIC     #^C7,R2         ;STRIP AWAY THE ASCII
3336  017624  005725                  TST     (R5)+           ;ADJUST FOR RETURN
3337  017626  000205          1$:     RTS     R5              ;RETURN
3338
3339                                  ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
3340                                  ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
3341                                  ;CALL
3342                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3343                                  ;       JSR     R5,CK.DEC       ;CHECK THE CHARACTER
3344                                  ;       RETURN1                 ;NOT BETWEEN 0 AND 9
3345                                  ;       RETURN2                 ;BETWEEN 0 AND 9
3346                                  ;                               ;R2 = DIGIT
3347
3348  017630  121127  000060  CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
3349  017634  103407                  BLO     1$              ;YES -- BRANCH
3350  017636  121127  000071          CMPB    (R1),#'9        ;GREATER THAN NINE?
3351  017642  101004                  BHI     1$              ;YES -- BRANCH
3352  017644  111102                  MOVB    (R1),R2         ;GET THE CHARACTER
3353  017646  042702  000060          BIC     #'0,R2          ;STRIP AWAY THE ASCII
3354  017652  005725                  TST     (R5)+           ;ADJUST FOR RETURN
3355  017654  000205          1$:     RTS     R5              ;RETURN
3356
3357                                  ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
3358                                  ;DETERMINE WHAT IT IS.
3359                                  ;CALL
3360                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3361                                  ;       JSR     R5,CK.CHR       ;CHECK CHARACTER
3362                                  ;       RETURN  ADR1            ;UNKNOWN CHARACTER
3363                                  ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
3364                                  ;       RETURN  ADR3            ;COMMA * (R1)=ADR+1
3365                                  ;       RETURN  ADR4            ;PERIOD * (R1)=ADR+1
3366                                  ;       RETURN  ADR5            ;DIGIT BETWEEN 0 AND 7.
3367                                  ;       RETURN  ADR6            ;DIGIT BETWEEN 8 AND 9.
3368                                  ;                               ;R2 = DIGIT * (R1)=ADR+1
3369
3370  017656  105711          CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
3371  017660  001417                  BEQ     3$              ;YES -- BRANCH
3372  017662  121127  000054          CMPB    (R1),#',        ;"COMMA"?
3373  017666  001413                  BEQ     2$              ;YES -- BRANCH
3374  017670  121127  000056          CMPB    (R1),#'.        ;"PERIOD"?
3375  017674  001407                  BEQ     1$              ;YES -- BRANCH
3376  017676  004537  017630          JSR     R5,CK.DEC       ;"DIGIT"?
3377  017702  000410                  BR      4$              ;NO -- BRANCH
3378  017704  004537  017602          JSR     R5,CK.OCT       ;OCTAL ?
3379  017710  005725                  TST     (R5)+           ;DIGIT BETWEEN 8-9
3380  017712  005725                  TST     (R5)+           ;DIGIT BETWEEN 0-7
3381  017714  005725          1$:     TST     (R5)+           ;PERIOD
3382  017716  005725          2$:     TST     (R5)+           ;COMMA
3383  017720  005725          3$:     TST     (R5)+           ;CARRIAGE RETURN
3384  017722  005201                  INC     R1              ;MOVE POINTER TO NEXT CHARACTER
3385  017724  011505          4$:     MOV     (R5),R5         ;UNKNOWN CHARACTER
```

```
3386  017726  000205                  RTS     R5              ;RETURN
3387
3388                                  ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
3389                                  ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
3390                                  ;CALL
3391                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
3392                                  ;       MOV     #NUM,R2         ;MAX. MAGNITUDE OF INPUT NUMBER
3393                                  ;       JSR     R5,CK.DIG       ;CHECK DIGITS
3394                                  ;       RETURN  ADR1            ;"CR" ONLY ENTERED -- R2=0
3395                                  ;       RETURN  ADR2            ;"PERIOD" ONLY ENTERED -- R2=0
3396                                  ;       RETURN  ADR3            ;ILLEGAL CHARACTER OR INPUT TOO LARGE -- R2=?
3397                                  ;       RETURN  ADR4            ;"CR" -- R2 = NUMBER
3398                                  ;       RETURN  ADR5            ;"COMMA" -- R2 = NUMBER
3399                                  ;       RETURN  ADR6            ;"PERIOD" -- R2 = NUMBER
3400
3401  017730  010446                  CK.DIG: MOV     R4,-(SP)        ;SAVE R4
3402  017732  010346                          MOV     R3,-(SP)        ;SAVE R3
3403  017734  010246                          MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
3404  017736  005002                          CLR     R2              ;START WITH 0
3405  017740  005003                          CLR     R3
3406  017742  005004                          CLR     R4
3407  017744  004537  017656                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
3408  017750  020044                          6$                      ;ILLEGAL CHARACTER
3409  017752  020052                          9$                      ;CARRIAGE RETURN
3410  017754  020044                          6$                      ;" "
3411  017756  020046                          7$                      ;"."
3412  017760  017764                          1$                      ;DIGIT 0-7
3413  017762  017764                          1$                      ;DIGIT 8-9
3414  017764  062705  000004          1$:     ADD     #4,R5           ;STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
3415  017770  006303                  2$:     ASL     R3              ;INPUT NUMBER *2
3416  017772  010346                          MOV     R3,-(SP)        ;SAVE *2
3417  017774  006303                          ASL     R3              ;*4
3418  017776  006303                          ASL     R3              ;*8
3419  020000  062603                          ADD     (SP)+,R3        ;(*2)+(*8) = *10
3420  020002  060203                          ADD     R2,R3           ;UPDATE THE INPUT NUMBER
3421  020004  004537  017656                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
3422  020010  020050                          8$                      ;ILLEGAL CHARACTER
3423  020012  020034                          5$                      ;CARRIAGE RETURN
3424  020014  020032                          4$                      ;" "
3425  020016  020024                          3$                      ;"."
3426  020020  017770                          2$                      ;DIGIT 0-7
3427  020022  017770                          2$                      ;DIGIT 8-9
3428  020024  105711                  3$:     TSTB    (R1)            ;DOES A "CR" FOLLOW THE "PERIOD"
3429  020026  001010                          BNE     8$              ;BR IF NOT
3430  020030  005724                          TST     (R4)+           ;INCREMENT THE RETURN
3431  020032  005724                  4$:     TST     (R4)+           ;INCREMENT THE RETURN
3432  020034  005724                  5$:     TST     (R4)+           ;INCREMENT THE RETURN
3433  020036  020316                          CMP     R3,(SP)         ;CHECK THE MAGNITUDE OF THE NUMBER
3434  020040  002004                          BGE     9$              ;BR IF ENTERED NUMBER TOO LARGE
3435  020042  000402                          BR      8$              ;BYPASS INCREMENT
3436  020044  005725                  6$:     TST     (R5)+           ;INCREMENT RETURN PAST INVALID RETURN
3437  020046  005725                  7$:     TST     (R5)+           ;INCREMENT RETURN
3438  020050  060405                  8$:     ADD     R4,R5           ;SETUP RETURN POINTER
3439  020052  010302                  9$:     MOV     R3,R2           ;ENTERED VALUE
3440  020054  005726                          TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
3441  020056  012603                          MOV     (SP)+,R3        ;RESTORE R3
```

```
                        MOV     (SP)+,R4        ;RESTORE R4
                        MOV     (R5),R5         ;GET RETURN ADDRESS
                        RTS     R5              ;RETURN

                .SBTTL  MACRO ROUTINES


                .SBTTL  ERROR HANDLER ROUTINE

                ;;*****************************************************************
                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                ;*AND GO TO TYPERR ON ERROR
                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                ;*SW15=1         HALT ON ERROR
                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                ;*SW10=1         BELL ON ERROR
                ;*SW09=1         LOOP ON ERROR
                ;*CALL
                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

                $ERROR:
                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
                        MOV     R1,DDRIVE       ;DRIVE ADDRESS IF DRIVE ERROR CALL
                        MOV     R3,ATTN         ;ATTENTION REGISTER CONTENTS
                        MOV     FMTDPB+12,DS.CYL  ;CURRENT CYLINDER ADDRESS
                        MOVB    FMTDPB+11,DS.TRK  ;REQUESTED TRACK ADDRESS
                        TST     RM.REG+RMBA     ;NON-ZERO BUFFER ADDRESS ?
                        BEQ     7$              ;BR IF NO BUFFER ADDRESS
                        MOV     RM.REG+RMBA,-(SP)  ;BUFFER ADDRESS
                        SUB     #2,(SP)         ;DECREMENT THE ADDRESS
                        MOV     @(SP)+,$GDDAT   ;GET THE BUFFER WORD WHICH DIDN'T COMPARE
                7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
                        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
                        BIT     #BIT10,@SWR     ;;BELL ON ERROR?
                        BEQ     1$              ;;NO - SKIP
                        TYPE    ,$BELL          ;;RING BELL
                1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
                        MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
                        SUB     #2,$ERRPC
                        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
                        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
                        BNE     20$             ;;SKIP TYPEOUTS
                        JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
                        TYPE    ,$CRLF
                20$:
                        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
                        BNE     2$              ;;NO,SKIP APT ERROR REPORT
                        MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
                        JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
                21$:    .BYTE   0
                        .BYTE   0
                22$:    BR      22$             ;;APT ERROR LOOP
                2$:     TST     @SWR            ;;HALT ON ERROR
                        BPL     3$              ;;SKIP IF CONTINUE
```

```
3498  020266  000000                        HALT                    ;;HALT ON ERROR!
3499  020270  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
3500  020272  032777  001000  160654  3$:   BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
3501  020300  001402                        BEQ     4$              ;;BR IF NO
3502  020302  013716  001124                MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
3503  020306  005737  001174        4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
3504  020312  001402                        BEQ     5$              ;;BR IF NONE
3505  020314  013716  001174                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
3506  020320                        5$:
3507  020320  000002                        RTI                     ;;RETURN
3508
3509
3510                                ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
3511                                ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
3512                                ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
3513                                ;CONCERNING THE ERROR.
3514
3515  020322  104412                TYPERR: SAVREG                  ;SAVE R0-R5
3516  020324  005000                        CLR     R0              ;CLEAR R0 FOR ERROR NUMBER
3517  020326  113700  001130                MOVB    $ITEMB,R0       ;ERROR NUMBER
3518  020332  005300                        DEC     R0              ;FORM INDEX FOR ERROR TABLE
3519  020334  006300                        ASL     R0
3520  020336  006300                        ASL     R0
3521  020340  006300                        ASL     R0
3522  020342  062700  006152        1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
3523  020346  012037  020362                MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
3524  020352  001404                        BEQ     3$              ;BRANCH IF THERE ISN'T ONE
3525  020354  104401  001203                TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
3526  020360  104401                        TYPE
3527  020362  000000        2$:     .WORD   0                       ;"EM" POINTER GOES HERE
3528  020364  012037  020400        3$:     MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
3529  020370  001404                        BEQ     5$              ;BRANCH IF NONE
3530  020372  104401  001203                TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3531  020376  104401                        TYPE
3532  020400  000000        4$:     .WORD   0                       ;"DH" POINTER GOES HERE
3533  020402  012001        5$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
3534  020404  001460                        BEQ     20$             ;BRANCH IF NONE
3535  020406  005005                        CLR     R5              ;SET INDENT SWITCH
3536  020410  012000                        MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
3537  020412  012002                        MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
3538  020414  001451                        BEQ     17$             ;BRANCH IF DH NUMBER IS 0
3539  020416  005105                        COM     R5              ;NO INDENT
3540  020420  104401  001203                TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3541  020424  112003        10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
3542  020426  112004                MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
3543  020430  006004        11$:    ROR     R4              ;OCTAL OR DECIMAL?
3544  020432  103403                BCS     12$             ;DECIMAL--BRANCH
3545  020434  013146                MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3546  020436  104402                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3547  020440  000402                BR      13$
3548  020442                12$:
3549  020442  013146                MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3550  020444  104405                TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3551  020446  005303        13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
3552  020450  001403                BEQ     14$             ;NO--BRANCH
3553  020452  104401  032503                TYPE    ,LINSP          ;YES--TYPE SEPERATORS
```

```
3554  020456  000764                    BR      11$             ;LOOP
3555  020460  005302                14$:DEC     R2              ;MORE DH'S?
3556  020462  003431                    BLE     20$             ;NO--BRANCH
3557  020464  104401  001203            TYPE    ,$CRLF          ;YES--START A NEW LINE
3558  020470  005760  000002            TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
3559  020474  001404                    BEQ     15$             ;BR IF YES - BYPASS THE INDENT
3560  020476  005105                    COM     R5              ;INDENT?
3561  020500  001002                    BNE     15$             ;NO--BRANCH
3562  020502  104401  032503            TYPE    ,LINSP          ;YES--TYPE SPACES
3563  020506  012037  020514        15$:MOV     (R0)+,16$       ;GET NEXT DH
3564  020512  104401                    TYPE                    ;AND TYPE IT
3565  020514  000000                16$:.WORD   0               ;DH POINTER GOES HERE
3566  020516  005710                    TST     (R0)            ;TYPE A 'DT' ?
3567  020520  001003                    BNE     21$             ;BR IF A 'DT'
3568  020522  062700  000004            ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
3569  020526  000754                    BR      14$             ;SEE IF END OF 'DF' BLOCK
3570  020530  104401  001203        21$:TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3571  020534  005705                    TST     R5              ;INDENT?
3572  020536  001332                    BNE     10$             ;NO--BRANCH
3573  020540  104401  032503        17$:TYPE    ,LINSP          ;YES--TYPE SPACES
3574  020544  000727                    BR      10$             ;LOOP
3575  020546  104413                20$:RESREG                  ;RESTORE R0-R5
3576  020550  000207                    RTS     PC              ;RETURN
3577
3578                                .SBTTL  TYPE ROUTINE
3579
3580                                ;;**************************************************************
3581                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3582                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3583                                ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3584                                ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3585                                ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3586                                ;*
3587                                ;*CALL:
3588                                ;*1) USING A TRAP INSTRUCTION
3589                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3590                                ;*OR
3591                                ;*      TYPE
3592                                ;*      MESADR
3593                                ;*
3594
3595  020552  105737  001173        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
3596  020556  100002                    BPL     1$              ;;BR IF YES
3597  020560  000000                    HALT                    ;;HALT HERE IF NO TERMINAL
3598  020562  000430                    BR      3$              ;;LEAVE
3599  020564  010046                1$: MOV     R0,-(SP)        ;;SAVE R0
3600  020566  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
3601  020572  122737  000001  001226    CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
3602  020600  001011                    BNE     6$              ;;NO,GO CHECK FOR APT CONSOLE
3603  020602  132737  000100  001227    BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
3604  020610  001405                    BEQ     6$              ;;NO,GO CHECK FOR CONSOLE
3605  020612  010037  020622            MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
3606  020616  004737  021042            JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
3607  020622  000000                61$:.WORD   0               ;;MESSAGE ADDRESS
3608  020624  132737  000040  001227  6$: BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
3609  020632  001003                    BNE     60$             ;;YES,SKIP TYPE OUT
```

TYPE ROUTINE

```
3610  020634  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3611  020636  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
3612  020640  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
3613  020642  012600                6$:     MOV     (SP)+,R0        ;;RESTORE R0
3614  020644  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
3615  020650  000002                        RTI                     ;;RETURN
3616  020652  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
3617  020656  001430                        BEQ     8$
3618  020660  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
3619  020664  001006                        BNE     5$
3620  020666  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
3621  020670  104401                        TYPE                    ;;TYPE A CR AND LF
3622  020672  001203                        $CRLF
3623  020674  105037  021030                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
3624  020700  000755                        BR      2$              ;;GET NEXT CHARACTER
3625  020702  004737  020764        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
3626  020706  123726  001172        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3627  020712  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
3628  020714  013746  001170                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3629                                                                ;;AND THE NULL CHAR.
3630  020720  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3631  020724  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3632  020726  004737  020764                JSR     PC,$TYPEC       ;;GO TYPE A NULL
3633  020732  105337  021030                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
3634  020736  000770                        BR      7$              ;;LOOP
3635
3636                                ;HORIZONTAL TAB PROCESSOR
3637
3638  020740  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3639  020744  004737  020764        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3640  020750  132737  000007  021030        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3641  020756  001372                        BNE     9$              ;;TAB STOP
3642  020760  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
3643  020762  000724                        BR      2$              ;;GET NEXT CHARACTER
3644  020764  105777  160174        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3645  020770  100375                        BPL     $TYPEC
3646  020772  116677  000002  160166        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3647  021000  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3648  021006  001003                        BNE     1$              ;;BRANCH IF NO
3649  021010  105037  021030                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3650  021014  000406                        BR      $TYPEX          ;;EXIT
3651  021016  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3652  021024  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
3653  021026  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
3654  021030  000000        $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
3655  021032  000207        $TYPEX: RTS     PC
3656
3657
3658
3659                                .SBTTL  APT COMMUNICATIONS ROUTINE
3660
3661                                ;;**************************************************************
3662  021034  112737  000001  021300  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
3663  021042  112737  000001  021276  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
3664  021050  000403                        BR      $ATYC
3665  021052  112737  000001  021300  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
```

```
3666  021060
3667  021060  010046                  $ATYC:  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3668  021062  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3669  021064  105737  021276                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
3670  021070  001450                          BEQ     5$              ;;IF NOT:  BR
3671  021072  122737  000001  001226        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
3672  021100  001031                          BNE     3$              ;;IF NOT:  BR
3673  021102  132737  000100  001227        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
3674  021110  001425                          BEQ     3$              ;;IF NOT:  BR
3675  021112  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
3676  021116  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3677  021124  005737  001206          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
3678  021130  001375                          BNE     1$              ;;IF NOT:  WAIT
3679  021132  010037  001222                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
3680  021136  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
3681  021140  001376                          BNE     2$
3682  021142  163700  001222                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
3683  021146  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
3684  021150  010037  001224                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
3685  021154  012737  000004  001206        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
3686  021162  000413                          BR      5$
3687  021164  017637  000004  021210  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
3688  021172  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
3689  021200  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
3690  021204  004737  020552                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
3691  021210  000000                  4$:     .WORD   0
3692  021212                          5$:
3693  021212  105737  021300          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
3694  021216  001416                          BEQ     12$             ;;IF NOT:  BR
3695  021220  005737  001226                  TST     $ENV            ;;RUNNING UNDER APT?
3696  021224  001413                          BEQ     12$             ;;IF NOT:  BR
3697  021226  005737  001206          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
3698  021232  001375                          BNE     11$             ;;IF NOT:  WAIT
3699  021234  017637  000004  001210        MOV     @4(SP),$FATAL   ;;GET ERROR #
3700  021242  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3701  021250  005237  001206                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
3702  021254  105037  021300          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
3703  021260  105037  021277                  CLRB    $LFLG           ;;CLEAR LOG FLAG
3704  021264  105037  021276                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
3705  021270  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3706  021272  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3707  021274  000207                          RTS     PC              ;;RETURN
3708  021276     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
3709  021277     000          $LFLG:  .BYTE   0               ;;LOG FLAG
3710  021300     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
3711        021302                          .EVEN
3712        000200          APTSIZE=200
3713        000001          APTENV=001
3714        000100          APTSPOOL=100
3715        000040          APTCSUP=040
3716
3717                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3718
3719                        ;;**************************************************************
3720                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3721                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
```

```
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT

021302  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
021306  116637  000001  021525        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
021314  112637  021527                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
021320  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
021324  000406                        BR      $TYPON
021326  112737  000001  021525  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
021334  112737  000006  021527        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
021342  112737  000005  021524  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
021350  010346                        MOV     R3,-(SP)        ;;SAVE R3
021352  010446                        MOV     R4,-(SP)        ;;SAVE R4
021354  010546                        MOV     R5,-(SP)        ;;SAVE R5
021356  113704  021527                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
021362  005404                        NEG     R4
021364  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
021370  110437  021526                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
021374  113704  021525                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
021400  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
021404  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
021406  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
021410  000404                        BR      3$              ;;GO DO MSB
021412  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
021414  006105                        ROL     R5
021416  006105                        ROL     R5
021420  010503                        MOV     R5,R3
021422  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
021424  105337  021526                DECB    $OMODE          ;;TYPE THIS DIGIT?
021430  100016                        BPL     7$              ;;BR IF NO
021432  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
021436  001002                        BNE     4$              ;;TEST FOR 0
021440  005704                        TST     R4              ;;SUPPRESS THIS 0?
021442  001403                        BEQ     5$              ;;BR IF YES
021444  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
021446  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
021452  052703  000040                BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
021456  110337  021522                MOVB    R3,8$           ;;SAVE FOR TYPING
021462  104401  021522                TYPE    ,8$             ;;GO TYPE THIS DIGIT
```

```
3778  021466  105337  021524          7$:     DECB    $OCNT           ;;COUNT BY 1
3779  021472  003347                          BGT     2$              ;;BR IF MORE TO DO
3780  021474  002402                          BLT     6$              ;;BR IF DONE
3781  021476  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3782  021500  000744                          BR      2$              ;;GO DO THE LAST DIGIT
3783  021502  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3784  021504  012604                          MOV     (SP)+,R4        ;;RESTORE R4
3785  021506  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3786  021510  016666  000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3787  021516  012616                          MOV     (SP)+,(SP)
3788  021520  000002                          RTI                     ;;RETURN
3789  021522     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3790  021523     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3791  021524     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3792  021525     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3793  021526  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3794
3795                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3796
3797                                  ;;*****************************************************************
3798                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3799                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3800                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3801                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3802                                  ;*REPLACED WITH SPACES.
3803                                  ;*CALL:
3804                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
3805                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
3806
3807  021530                          $TYPDS:
3808  021530  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3809  021532  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3810  021534  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3811  021536  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3812  021540  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3813  021542  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
3814  021546  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
3815  021552  100004                          BPL     1$              ;;BR IF INPUT IS POS.
3816  021554  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
3817  021556  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
3818  021564  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
3819  021566  012703  021744                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
3820  021572  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
3821  021576  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
3822  021600  016001  021734                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
3823  021604  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
3824  021606  002402                          BLT     4$              ;;BR IF DONE
3825  021610  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
3826  021612  000774                          BR      3$
3827  021614  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
3828  021616  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
3829  021620  001002                          BNE     5$              ;;FALL THROUGH IF 0
3830  021622  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
3831  021624  100407                          BMI     7$              ;;BR IF YES
3832  021626  106316                  5$:     ASLB    (SP)            ;;MSD?
3833  021630  103003                          BCC     6$              ;;BR IF NO
```

```
3834  021632  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
3835  021640  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
3836  021644  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
3837  021650  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
3838  021652  005720                          TST     (R0)+           ;;JUST INCREMENTING
3839  021654  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
3840  021660  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
3841  021662  003002                          BGT     8$              ;;GO TO EXIT
3842  021664  010502                          MOV     R5,R2           ;;GET THE LSD
3843  021666  000764                          BR      6$              ;;GO CHANGE TO ASCII
3844  021670  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
3845  021672  100003                          BPL     9$              ;;BR IF NO
3846  021674  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
3847  021702  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
3848  021704  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
3849  021706  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
3850  021710  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
3851  021712  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3852  021714  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3853  021716  104401  021744                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
3854  021722  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
3855  021730  012616                          MOV     (SP)+,(SP)
3856  021732  000002                          RTI                     ;;RETURN TO USER
3857  021734  023420                  $DTBL:  10000.
3858  021736  001750                          1000.
3859  021740  000144                          100.
3860  021742  000012                          10.
3861  021744  000004                  $DBLK:  .BLKW   4
3862
3863                                  .SBTTL  TTY INPUT ROUTINE
3864
3865                                  ;;*****************************************************************
3866                                  .ENABL  LSB
3867  021754  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
3868  021756  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
3869  021760  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
3870  021762  000007                  $TKQSRT: .BLKB  7               ;;TTY KEYBOARD QUEUE
3871          021771                  $TKQEND=.
3872          021772                  .EVEN
3873
3874                                  ;*TK INITIALIZE ROUTINE
3875                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
3876                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
3877                                  ;
3878                                  ;*CALL:
3879                                  ;*      JSR     PC,$TKINT
3880                                  ;*      RETURN
3881                                  ;
3882  021772  005037  021754          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
3883  021776  012737  021762  021756          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
3884  022004  013737  021756  021760          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
3885  022012  012737  022042  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
3886  022020  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
3887  022026  005777  157130                  TST     @$TKB           ;;CLEAR DONE FLAG
3888  022032  012777  000100  157120          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
3889  022040  000207                          RTS     PC              ;;RETURN TO CALLER
```

```
                                        ;*TK SERVICE ROUTINE
                                        ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                        ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                        ;*IT IN THE QUEUE.
                                        ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
                                        ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (@CNTLC)
                                        ;
022042  117746  157114          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
022046  042716  177600                  BIC     #↑C177,(SP)     ;;STRIP THE JUNK
022052  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
022056  001007                          BNE     1$              ;;BRANCH IF NO
022060  104401  023163                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
022064  004737  021772                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
022070  005726                          TST     (SP)+           ;;CLEAN UP STACK
022072  000177  157300                  JMP     @CNTLC          ;;CONTROL C RESTART
022076  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
022102  001004                          BNE     2$              ;;BRANCH IF NO
022104  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
022112  001500                          BEQ     6$              ;;GO TO SWR CHANGE
                                        ;
022114                          2$:
022114  022737  000007  021754          CMP     #7,$TKCNT       ;;IS THE QUEUE FULL?
022122  001004                          BNE     3$              ;;BRANCH IF NO
022124  104401  001176                  TYPE    ,$BELL          ;;RING THE TTY BELL
022130  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
022132  000451                          BR      5$              ;;EXIT
022134  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
022140  001021                          BNE     32$             ;;BRANCH IF NO
022142  005077  157012                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
022146  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
022150  105777  157004          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
022154  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
022156  117746  157000                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
022162  042716  177600                  BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
022166  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
022172  001366                          BNE     31$             ;;BRANCH IF NO
022174  012777  000100  156756          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
022202  000002                          RTI                     ;;RETURN
022204  005237  021754          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
022210  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
022214  002405                          BLT     4$              ;;BRANCH IF YES
022216  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
022222  003002                          BGT     4$              ;;BRANCH IF YES
022224  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
022230  112677  177522          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
022234  005237  021756                  INC     $TKQIN          ;;UPDATE THE POINTER
022240  023727  021756  021771          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
022246  001003                          BNE     5$              ;;BRANCH IF NO
022250  012737  021762  021756          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
022256  000002                  5$:     RTI                     ;;RETURN
                                        ;
                                        ;;*****************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
```

```
                                        ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
022260  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
022266  001124                   BNE     15$             ;;EXIT IF NOT
022270  105777  156664           TSTB    @$TKS           ;;IS A CHAR WAITING?
022274  100121                   BPL     15$             ;;IF NOT, EXIT
022276  117746  156660           MOVB    @$TKB,-(SP)     ;;YES
022302  042716  177600           BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
022306  021627  000007           CMP     (SP),#7         ;;IS IT A CONTROL-G?
022312  001300                   BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                         ;;AND EXIT


                                        ;;*****************************************************************
                                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
022314  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
022322  001674                   BEQ     2$              ;;BRANCH IF YES
022324  005726                   TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
022326  004737  021772           JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
022332  005077  156622           CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
022336  112737  000001  001151   MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

022344  104401  023175           TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
022350  104401  023202  $GTSWR:  TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
022354  013746  000176           MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
022360  104402                   TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
022362  104401  023213           TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
022366  005046           19$:    CLR     -(SP)           ;;CLEAR COUNTER
022370  005046                   CLR     -(SP)           ;;THE NEW SWR
022372  105777  156562   7$:     TSTB    @$TKS           ;;CHAR THERE?
022376  100375                   BPL     7$              ;;IF NOT TRY AGAIN

022400  117746  156556           MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
022404  042716  177600           BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

022410  021627  000003           CMP     (SP),#3         ;;IS IT A CONTROL-C?
022414  001015                   BNE     9$              ;;BRANCH IF NOT
022416  104401  023163           TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
022422  062706  000006           ADD     #6,SP           ;;CLEAN UP STACK
022426  123727  001151  000001   CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
022434  001003                   BNE     8$              ;;BRANCH IF NO
022436  012777  000100  156514   MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
022444  000177  156726   8$:     JMP     @CNTLC          ;;CONTROL-C RESTART

022450  021627  000025   9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
022454  001005                   BNE     10$             ;;BRANCH IF NOT
022456  104401  023170           TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
022462  062706  000006   20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
022466  000737                   BR      19$             ;;LET'S TRY IT AGAIN

022470  021627  000015   10$:    CMP     (SP),#15        ;;IS IT A <CR>?
022474  001022                   BNE     16$             ;;BRANCH IF NO
022476  005766  000004           TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
022502  001403                   BEQ     11$             ;;BRANCH IF YES
```

```
4002  022504  016677  000002  156442          MOV     2(SP),@SWR      ;;SAVE NEW SWR
4003  022512  062706  000006                  ADD     #6,SP           ;;CLEAR UP STACK
4004  022516  104401  001203          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
4005  022522  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
4006  022530  001003                          BNE     15$             ;;BRANCH IF NOT
4007  022532  012777  000100  156420          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4008  022540  000002          15$:            RTI                     ;;RETURN
4009  022542  004737  020764  16$:            JSR     PC,$TYPEC       ;;ECHO CHAR
4010  022546  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
4011  022552  002420                          BLT     18$             ;;BRANCH IF YES
4012  022554  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
4013  022560  003015                          BGT     18$             ;;BRANCH IF YES
4014  022562  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
4015  022566  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
4016  022572  001403                          BEQ     17$             ;;BRANCH IF YES
4017  022574  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
4018  022576  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
4019  022600  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
4020  022602  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
4021  022606  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
4022  022612  000667                          BR      7$              ;;GET THE NEXT ONE
4023  022614  104401  001202          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
4024  022620  000720                          BR      20$             ;;SIMULATE CONTROL-U
4025                                  .DSABL  LSB
4026
4027
4028                                  ;;**************************************************************
4029                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4030                                  ;*CALL:
4031                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
4032                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
4033                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
4034                                  ;
4035
4036  022622  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
4037  022624  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
4038  022632  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
4039  022636  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
4040  022640  012746  022646                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
4041  022644  000002                          RTI                     ;;POP NEW PC AND PS
4042  022646                          64$:
4043  022646  005737  021754          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
4044  022652  001775                          BEQ     1$
4045  022654  005337  021754                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
4046  022660  117766  177074  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
4047  022666  005237  021760                  INC     $TKQOUT         ;;UPDATE THE POINTER
4048  022672  023727  021760  021771          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
4049  022700  001003                          BNE     2$              ;;BRANCH IF NO
4050  022702  012737  021762  021760          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
4051  022710  000002          2$:             RTI                     ;;RETURN
4052                                  ;;**************************************************************
4053                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4054                                  ;*CALL:
4055                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
4056                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4057                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
```

```
                                ;SRDLIN: MOV     R3,-(SP)        ;;SAVE R3
                                         CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
                                ;1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
                                ;2$:     CMP     #$TTYIN+15,R3   ;;BUFFER FULL?
                                         BLOS    4$              ;;BR IF YES
                                         RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
                                         MOVB    (SP)+,(R3)      ;;GET CHARACTER
                                ;10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
                                         BNE     5$              ;;BR IF NO
                                         TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
                                         BNE     6$              ;;BR IF NO
                                         MOVB    #'\,-(SP)       ;;TYPE A BACK SLASH
                                         TYPE    ,-(SP)
                                         MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
                                ;6$:     DEC     R3              ;;BACKUP BY ONE
                                         CMP     R3,#$TTYIN      ;;STACK EMPTY?
                                         BLO     4$              ;;BR IF YES
                                         MOVB    (R3),-(SP)      ;;SETUP TO TYPEOUT THE DELETED CHAR.
                                         TYPE    ,(SP)+          ;;GO TYPE
                                         BR      2$              ;;GO READ ANOTHER CHAR.
                                ;5$:     TST     (SP)            ;;RUBOUT KEY SET?
                                         BEQ     7$              ;;BR IF NO
                                         MOVB    #'\,-(SP)       ;;TYPE A BACK SLASH
                                         TYPE    ,-(SP)
                                         CLR     (SP)+           ;;CLEAR THE RUBOUT KEY
                                ;7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
                                         BNE     8$              ;;BR IF NO
                                         TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
                                         BR      1$              ;;GO START OVER
                                ;8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
                                         BNE     3$              ;;BRANCH IF NO
                                         CLRB    (R3)            ;;CLEAR THE CHARACTER
                                         TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
                                         TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
                                         BR      2$              ;;GO PICKUP ANOTHER CHACTER
                                ;4$:     TYPE    ,$QUES          ;;TYPE A '?'
                                         BR      1$              ;;CLEAR THE BUFFER AND LOOP
                                ;3$:     MOVB    (R3),-(SP)      ;;ECHO THE CHARACTER
                                         TYPE    ,(SP)+
                                         CMPB    #15,(R3)+       ;;CHECK FOR RETURN
                                         BNE     2$              ;;LOOP IF NOT RETURN
                                         CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
                                         TYPE    ,$LF            ;;TYPE A LINE FEED
                                         TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
                                         MOV     (SP)+,R3        ;;RESTORE R3
                                         MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
                                         MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
                                         MOV     #$TTYIN,4(SP)
                                         RTI                     ;;RETURN
                                ;9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
                                         .BYTE   0               ;;TERMINATOR
                                ;$TTYIN: .BLKB   15              ;;RESERVE 15 BYTES FOR TTY INPUT
                                ;$CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
                                ;$CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
                                ;$CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
```

```
4114  023202  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
4115  023210  020075     000
4116  023213     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
4117  023220  036440  000040
4118
4119                                  .SBTTL  POWER DOWN AND UP ROUTINES
4120
4121                                  ;;*****************************************************************
4122                                  ;POWER DOWN ROUTINE
4123  023224  012737  023370  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
4124  023232  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
4125  023240  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4126  023242  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4127  023244  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4128  023246  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4129  023250  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
4130  023252  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
4131  023254  017746  155674                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
4132  023260  010637  023374                MOV     SP,$SAVR6       ;;SAVE SP
4133  023264  012737  023276  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
4134  023272  000000                        HALT
4135  023274  000776                        BR      .-2             ;;HANG UP
4136
4137                                  ;;*****************************************************************
4138                                  ;POWER UP ROUTINE
4139  023276  012737  023370  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
4140  023304  013706  023374                MOV     $SAVR6,SP       ;;GET SP
4141  023310  005037  023374                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
4142  023314  005237  023374          1$:   INC     $SAVR6          ;;WAIT FOR THE INC
4143  023320  001375                        BNE     1$              ;;OF  WORD
4144  023322  012677  155626                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
4145  023326  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
4146  023330  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
4147  023332  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
4148  023334  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
4149  023336  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
4150  023340  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
4151  023342  012737  023224  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
4152  023350  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
4153  023356  104401                        TYPE                    ;;REPORT THE POWER FAILURE
4154  023360  033734          $PWRMG: .WORD   PMSG                    ;;POWER FAIL MESSAGE POINTER
4155  023362  012716                        MOV     (PC)+,(SP)      ;;RESTART AT BEGIN2
4156  023364  006436          $PWRAD: .WORD   BEGIN2                  ;;RESTART ADDRESS
4157  023366  000002                        RTI
4158  023370  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
4159  023372  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
4160  023374  000000          $SAVR6: 0                               ;;PUT THE SP HERE
4161
4162                                  .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
4163
4164                                  ;;*****************************************************************
4165                                  ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
4166                                  ;*LEADING NUMBERS.
4167                                  ;*CALL
4168                                  ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
4169                                  ;*      JSR     PC,@#$SUPRS
```

```
                                ;SUPPRESS LEADING ZEROS
023376  010046          $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
023400  016600  000004          MOV     4(SP),R0        ;;PICKUP THE POINTER
023404  105710          1$:     TSTB    (R0)            ;;TERMINATEOR?
023406  001403                  BEQ     2$              ;;BR IF YES
023410  122720  000060          CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
023414  001773                  BEQ     1$              ;;BR IF YES
023416  005300          2$:     DEC     R0              ;;BACKUP BY "1"
023420  010037  023426          MOV     R0,3$           ;;SAVE FOR TYPING
023424  104401                  TYPE                    ;;GO TYPE
023426  000000          3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
023430  012600                  MOV     (SP)+,R0        ;;RESTORE R0
023432  012616                  MOV     (SP)+,(SP)      ;;RESTORE THE STACK
023434  000207                  RTS     PC              ;;RETURN

                                .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

                                ;;**************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                                ;*CALL
                                ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                ;*      JSR     PC,@#$SB2D      ;;CALL
                                ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK


023436  016637  000002  023466  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
023444  012746  023466          MOV     #1$,-(SP)       ;;SET POINTER
023450  004737  023472          JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
023454  062716  000005          ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
023460  012666  000002          MOV     (SP)+,2(SP)     ;;PICKUP POINTER
023464  000207                  RTS     PC              ;;RETURN
023466  000000  000000  1$:     .WORD   0,0

                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                ;;**************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                ;*POSITIVE.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB2D
                                ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                ;;IS ON THE STACK


023472  104412          $DB2D:  SAVREG                  ;;SAVE REGISTERS
023474  016602  000002          MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
023500  012700  023652          MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
023504  010066  000002          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
023510  012201                  MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
023512  012202                  MOV     (R2)+,R2
023514  012737  000012  023570  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
023522  012704  023602          MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
```

```
4226  023526  012705  023604          MOV     #$TNPWR+2,R5
4227  023532  005003          1$:     CLR     R3              ;;CLEAR PARTIAL
4228  023534  161401          2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
4229  023536  005602                  SBC     R2
4230  023540  161502                  SUB     (R5),R2
4231  023542  002402                  BLT     3$              ;;BR IF TEN POWER TO LARGE
4232  023544  005203                  INC     R3              ;;ADD 1 TO PARTIAL
4233  023546  000772                  BR      2$              ;;LOOP
4234  023550  062401          3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
4235  023552  005502                  ADC     R2
4236  023554  062402                  ADD     (R4)+,R2
4237  023556  022525                  CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
4238  023560  052703  000060          BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
4239  023564  110320                  MOVB    R3,(R0)+        ;;SAVE IT
4240  023566  005327                  DEC     (PC)+           ;;DONE?
4241  023570  000000          4$:     .WORD   0
4242  023572  001357                  BNE     1$              ;;BR IF NO
4243  023574  105020                  CLRB    (R0)+           ;;TERMINATOR
4244  023576  104413                  RESREG                  ;;RESTORE REGISTERS
4245  023600  000207                  RTS     PC              ;;RETURN
4246  023602  145000          $TNPWR: 145000                  ;;1.0E09
4247  023604  035632                  35632
4248  023606  160400                  160400                  ;;1.0E08
4249  023610  002765                  2765
4250  023612  113200                  113200                  ;;1.0E07
4251  023614  000230                  230
4252  023616  041100                  041100                  ;;1.0E06
4253  023620  000017                  17
4254  023622  103240                  103240                  ;;1.0E05
4255  023624  000001                  1
4256  023626  023420                  23420                   ;;1.0E04
4257  023630  000000                  0
4258  023632  001750                  1750                    ;;1.0E03
4259  023634  000000                  0
4260  023636  000144                  144                     ;;1.0E02
4261  023640  000000                  0
4262  023642  000012                  12                      ;;1.0E01
4263  023644  000000                  0
4264  023646  000001                  1                       ;;1.0E00
4265  023650  000000                  0
4266  023652  000014  $DECVL: .BLKB   12.                     ;;RESERVE STORAGE FOR ASCIZ STRING
4267
4268                    .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
4269
4270                    ;;*****************************************************************
4271                    ;*SAVE R0-R5
4272                    ;*CALL:
4273                    ;*      SAVREG
4274                    ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
4275                    ;*
4276                    ;*TOP---(+16)
4277                    ;* +2---(+18)
4278                    ;* +4---R5
4279                    ;* +6---R4
4280                    ;* +8---R3
4281                    ;*+10---R2
```

```
                                ;*+12---R1
                                ;*+14---R0

                                $SAVREG:
023666                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
023666  010046                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
023670  010146                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
023672  010246                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
023674  010346                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
023676  010446                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
023700  010546                         MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
023702  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
023706  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
023712  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
023716  016646  000022                 RTI
023722  000002

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
023724                          $RESREG:
023724  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
023730  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
023734  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
023740  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
023744  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
023746  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
023750  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
023752  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
023754  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
023756  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
023760  000002                          RTI

                                .SBTTL  TRAP DECODER

                                ;;**************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

023762  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
023764  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
023770  005740                          TST     -(R0)           ;;BACKUP BY 2
023772  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
023774  006300                          ASL     R0              ;;POSITION FOR INDEXING
023776  016000  024016                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
024002  000200                          RTS     R0              ;;GO TO ROUTINE

                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

024004  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
024006  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
024014  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE
```

```
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
024016  024004          $TRPAD: .WORD   $TRAP2
024020  020552                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
024022  021326                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
024024  021302                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
024026  021342                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
024030  021530                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

024032  022350                  $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

024034  022260                  $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
024036  022622                  $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
024040  022712                  $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
024042  023666                  $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
024044  023724                  $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE


                                .SBTTL  SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

                                ;COPYRIGHT (C) 1976
                                ;DIGITAL EQUIPMENT CORP.
                                ;MAYNARD, MA 01754
                                ;AUTHOR($): JIM LACEY/CHUCK HESS/C. CHEN

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

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

024046  000000  000000  000000  RMERRS: .WORD   0,0,0,0
024054  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

024056      000         DRVACT: .BYTE   0               ;DRIVE 0
024057      000                 .BYTE   0               ;DRIVE 1
024060      000                 .BYTE   0               ;DRIVE 2
024061      000                 .BYTE   0               ;DRIVE 3
024062      000                 .BYTE   0               ;DRIVE 4
024063      000                 .BYTE   0               ;DRIVE 5
024064      000                 .BYTE   0               ;DRIVE 6
024065      000                 .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
```

```
                                ;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXSITENT
                                ;DRVSTA>0 IF DRIVE IS ONLINE
                                ;DRVSTA<0 IF DRIVE IS UNSAFE

024066  000             DRVSTA: .BYTE   0               ;DRIVE 0
024067  000                     .BYTE   0               ;DRIVE 1
024070  000                     .BYTE   0               ;DRIVE 2
024071  000                     .BYTE   0               ;DRIVE 3
024072  000                     .BYTE   0               ;DRIVE 4
024073  000                     .BYTE   0               ;DRIVE 5
024074  000                     .BYTE   0               ;DRIVE 6
024075  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 RM03
                                ;DRVTYP=-1 IF NOT RM03

024076  000             DRVTYP: .BYTE   0               ;DRIVE 0
024077  000                     .BYTE   0               ;DRIVE 1
024100  000                     .BYTE   0               ;DRIVE 2
024101  000                     .BYTE   0               ;DRIVE 3
024102  000                     .BYTE   0               ;DRIVE 4
024103  000                     .BYTE   0               ;DRIVE 5
024104  000                     .BYTE   0               ;DRIVE 6
024105  000                     .BYTE   0               ;DRIVE 7

                        ;TABLE OF DUAL PORT INITIALIZATION INDICATORS
                                ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
                                ;DPINT<0 IF INITIALIZATION IS IN PROGRESS

024106  000             DPINT:  .BYTE   0               ;DRIVE 0
024107  000                     .BYTE   0               ;DRIVE 1
024110  000                     .BYTE   0               ;DRIVE 2
024111  000                     .BYTE   0               ;DRIVE 3
024112  000                     .BYTE   0               ;DRIVE 4
024113  000                     .BYTE   0               ;DRIVE 5
024114  000                     .BYTE   0               ;DRIVE 6
024115  000                     .BYTE   0               ;DRIVE 7

                        ;TABLE OF PENDING DUAL PORT REQUESTS
                                ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
                                ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE

024116  000             DPRQS:  .BYTE   0               ;DRIVE 0
024117  000                     .BYTE   0               ;DRIVE 1
024120  000                     .BYTE   0               ;DRIVE 2
024121  000                     .BYTE   0               ;DRIVE 3
024122  000                     .BYTE   0               ;DRIVE 4
024123  000                     .BYTE   0               ;DRIVE 5
024124  000                     .BYTE   0               ;DRIVE 6
024125  000                     .BYTE   0               ;DRIVE 7

                        ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
                                ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
                                ;"DPB" OF THE I/O OPERATION.
```

```
                                ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
                                        ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
                                        ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
                                        ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
                                        ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.

                                ;RM03 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
                                        ;ACTDRV=0 IF DRIVER IS INACTIVE
                                        ;ACTDRV>0 IF DRIVER IS ACTIVE

                                ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
                                        ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
                                        ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE

                                ;UNLOAD FLAG (ULDFLG=8 BYTES)
                                        ;ULDFLG=0 IF NO UNLOAD COMMAND
                                        ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
                                        ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE

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

                                ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
                                        ;SAVEFG <0 IF SAVE THE RH70/RM03 REGISTERS WHEN THE
                                        ;OPERATION IS COMPLETED AS PER (DPB+14).
                                        ;SAVEFG=0 IF SAVE THE RH70/RM03 REGISTERS, AS PER
                                        ;(DPB+14), AFTER AN ERROR.
```

```
4451  024126  000000  TRNSWT: .WORD   0

4459  024130  000000  SRCHWT: .WORD   0

4465  024132     000  ACTDRV: .BYTE   0

4471  024133     000  ACTSTR: .BYTE   0

4478  024134     000  ULDFLG: .BYTE   0               ;DRIVE 0
4479  024135     000          .BYTE   0               ;DRIVE 1
4480  024136     000          .BYTE   0               ;DRIVE 2
4481  024137     000          .BYTE   0               ;DRIVE 3
4482  024140     000          .BYTE   0               ;DRIVE 4
4483  024141     000          .BYTE   0               ;DRIVE 5
4484  024142     000          .BYTE   0               ;DRIVE 6
4485  024143     000          .BYTE   0               ;DRIVE 7

4490  024144     000  LACNT:  .BYTE   0               ;DRIVE 0
4491  024145     000          .BYTE   0               ;DRIVE 1
4492  024146     000          .BYTE   0               ;DRIVE 2
4493  024147     000          .BYTE   0               ;DRIVE 3
4494  024150     000          .BYTE   0               ;DRIVE 4
4495  024151     000          .BYTE   0               ;DRIVE 5
4496  024152     000          .BYTE   0               ;DRIVE 6
4497  024153     000          .BYTE   0               ;DRIVE 7

4505  024154  000000  SAVEFG: .WORD   0
```

```
4506
4507                                    ;SEEK FLAG (SEEKFG=1 WORD)
4508                                            ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
4509                                            ;FOR A DATA TRANSFER START A SEARCH COMMAND
4510                                            ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
4511                                            ;DISREGARD THE WINDOW
4512
4513  024156  177777                    SEEKFG: .WORD   -1
4514
4515                                    ;TIMEOUT TABLE (TIMER=8 WORDS)
4516                                            ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
4517
4518  024160  177777                    TIMER:  .WORD   -1      ;DRIVE 0
4519  024162  177777                            .WORD   -1      ;DRIVE 1
4520  024164  177777                            .WORD   -1      ;DRIVE 2
4521  024166  177777                            .WORD   -1      ;DRIVE 3
4522  024170  177777                            .WORD   -1      ;DRIVE 4
4523  024172  177777                            .WORD   -1      ;DRIVE 5
4524  024174  177777                            .WORD   -1      ;DRIVE 6
4525  024176  177777                            .WORD   -1      ;DRIVE 7
4526
4527                                    ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
4528                                            ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
4529                                            ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
4530
4531  024200  177777                    DTUW:   .WORD   -1
4532
4533                                    ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
4534                                            ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
4535                                            ;ATTENTION BIT
4536
4537  024202      001                   ATABIT: .BYTE   1       ;DRIVE 0
4538  024203      002                           .BYTE   2       ;DRIVE 1
4539  024204      004                           .BYTE   4       ;DRIVE 2
4540  024205      010                           .BYTE   10      ;DRIVE 3
4541  024206      020                           .BYTE   20      ;DRIVE 4
4542  024207      040                           .BYTE   40      ;DRIVE 5
4543  024210      100                           .BYTE   100     ;DRIVE 6
4544  024211      200                           .BYTE   200     ;DRIVE 7
4545
4546                                    ;FSRM03 TO RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
4547                                    ;CALLING IT FATAL (MCPEMX=1 WORD)
4548
4549  024212  000003                    MCPEMX: .WORD   3
4550
4551                                    ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH70/RM03),
4552                                    ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).
4553
4554  024214  176700                    RMADR:  .WORD   176700
4555  024216  000254  000240            RMVEC:  .WORD   254,5*32.
4556
4557                                    ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
4558
4559  024222  000004                    MXLACT: .WORD   4
4560                                    ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
4561
```

```
4562  024224  001000                  MXDLTA: .WORD   8.*64.
4563                                  ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
4564
4565  024226  000200                  MNDLTA: .WORD   2*64.
4566                                  ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
4567
4568  024230  000005                  MXWNDW: .WORD   5
4569
4570                                  ;DEFINITIONS OF THE RH70/RM03 ADDRESS INDEXES
4571
4572          000000                  RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
4573          000002                  RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
4574          000004                  RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
4575          000006                  RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
4576          000010                  RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
4577          000012                  RMDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
4578          000014                  RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
4579          000016                  RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
4580          000020                  RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
4581          000022                  RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
4582          000024                  RMMR1=24                ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
4583          000026                  RMDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
4584          000030                  RMSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
4585          000032                  RMOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
4586          000034                  RMDC=34                 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
4587          000036                  RMHR=36                 ;DUMMY  ADDRESS REGISTER (DRIVE REG. 13)
4588          000040                  RMMR2=40                ;MAINTENANCE REGISTER #2
4589          000042                  RMER2=42                ;ERROR REGISTER #2 (DRIVE REG. 15)
4590          000044                  RMEC1=44                ;ECC POSITION REGISTER (DRIVE REG. 16)
4591          000046                  RMEC2=46                ;ECC PATTERN REGISTER (DRIVE REG.  17)
4592
4593                                  ;RH70/RM03 DRIVER INITIALIZATION CODE
4594                                          ;THIS ROUTINE WILL DETERMINE WHICH RM03 DRIVES ARE
4595                                          ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
4596                                          ;TO THE PROPER STATE FOR EACH DRIVE.
4597                                          ;NOTE: THIS ROUTINE CALLS DRVINT
4598                                  ;
4599                                  ;CALL
4600                                  ;
4601                                  ;       JSR     PC,RMINIT
4602                                  ;       RETURN
4603                                  ;
4604                                  ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
4605                                  ;
4606  024232  104412                  RMINIT: SAVREG                  ;SAVE R0 - R5
4607  024234  013746  177776                  MOV     @#PS,-(SP)     ;SAVE THE PRESENT PROCESSOR STATUS
4608  024240  012737  000240  177776          MOV     #<5*32.>,@#PS  ;CHANGE THE PRIORITY TO 5
4609  024246  004737  032046                  JSR     PC,CLRQUE      ;CLEAR ALL REQUEST QUEUES
4610  024252  012701  024046                  MOV     #RMERRS,R1     ;FIRST ADDRESS TO BE CLEARED
4611  024256  012702  024156                  MOV     #SEEKFG,R2     ;LAST ADDRESS TO BE CLEARED
4612  024262  005021                  1$:     CLR     (R1)+          ;CLEAR
4613  024264  020102                          CMP     R1,R2          ;ARE WE DONE?
4614  024266  103775                          BLO     1$             ;BRANCH IF NO
4615  024270  012702  024200                  MOV     #DTUW,R2       ;LAST ADDRESS
4616  024274  012721  177777          2$:     MOV     #-1,(R1)+      ;INITIALIZE
4617  024300  020102                          CMP     R1,R2          ;DONE?
```

```
4618  024302  101774                        BLOS    2$              ;LOOP IF NO
4619  024304  005037  024066                CLR     DRVSTA          ;SET ALL DRIVES TO OFFLINE
4620  024310  005037  024070                CLR     DRVSTA+2
4621  024314  005037  024072                CLR     DRVSTA+4
4622  024320  005037  024074                CLR     DRVSTA+6
4623  024324  013703  024216                MOV     RMVEC,R3        ;SETUP THE RH70/RM03 VECTOR
4624  024330  012723  027020                MOV     #ISR,(R3)+
4625  024334  013713  024220                MOV     RMVEC+2,(R3)
4626  024340  013704  024214                MOV     RMADR,R4        ;FIRST ADDRESS OF RH70/RM03
4627  024344  012764  000040  000010        MOV     #BIT05,RMCS2(R4)  ;MASSBUS INIT
4628  024352  005001                        CLR     R1              ;START WITH DRIVE 0
4629  024354  004037  024444          3$:   JSR     R0,DRVINT       ;INIT THE DRIVE
4630  024360  000401                        BR      4$              ;'DVA' NOT SET OR PARITY ERROR
4631  024362  000402                        BR      5$              ;NORMAL RETURN
4632  024364  105061  024066          4$:   CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
4633  024370  005201                5$:     INC     R1              ;GO TO NEXT DRIVE
4634  024372  042701  177770                BIC     #^C7,R1         ;MASK OUT UNUSED BITS
4635  024376  001366                        BNE     3$              ;BR IF MORE DRIVES TO GO
4636  024400  012701  000007                MOV     #7,R1           ;START WITH DRIVE 7
4637  024404  005037  177776                CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
4638  024410  105761  024106          6$:   TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
4639  024414  001405                        BEQ     8$              ;BR NOT WAITING
4640  024416  004737  031502                JSR     PC,SET.IE       ;SET INTERRUPT
4641  024422  105761  024106          7$:   TSTB    DPINT(R1)       ;DRIVE SWITCHED PORTS ?
4642  024426  001375                        BNE     7$              ;BR IF NOT
4643  024430  005301                8$:     DEC     R1              ;GO TO THE NEXT DRIVE
4644  024432  100366                        BPL     6$              ;CHECK NEXT DRIVE
4645  024434  012637  177776                MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
4646  024440  104413                        RESREG                  ;RESTORE R0 - R5
4647  024442  000207                        RTS     PC              ;BYE-BYE
4648
4649                                ;DRIVE INITILIZATION ROUTINE
4650                                        ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
4651                                        ;AN RM03. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
4652                                        ;IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
4653                                        ;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
4654                                        ;DRVSTA IS SET TO THE PROPER CONDITION.
4655                                ;CALL
4656                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
4657                                ;       MOV     RMADR,R4        ;UNIBUS ADDRESS OF RH70/RM03 (RMCS1)
4658                                ;       JSR     R0,DRVINT       ;CALLED BY A JSR
4659                                ;       RETURN1                 ;ERROR OCCURRED (PARITY)
4660                                ;       RETURN2                 ;NORMAL RETURN
4661                                ;
4662
4663  024444  010546                DRVINT: MOV     R5,-(SP)        ;SAVE R5
4664  024446  105061  024066        DULP:   CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
4665  024452  105061  024076                CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
4666  024456  105061  024134                CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
4667  024462  010164  000010                MOV     R1,RMCS2(R4)    ;SELECT A DRIVE
4668  024466  112764  000111  000000        MOVB    #111,RMCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
4669  024474  032764  010000  000010        BIT     #BIT12,RMCS2(R4)  ;NONEXISTENT DRIVE?
4670  024502  001403                        BEQ     1$              ;NO---BRANCH
4671  024504  004737  031502                JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
4672  024510  000476                        BR      6$              ;LEAVE THIS ROUTINE
4673  024512  105061  024066          1$:   CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
```

```
4674  024516  032764  004000  000000          BIT     #BIT11,RMCS1(R4)  ;SEE IF DRIVE AVAILABLE
4675  024524  001750                          BEQ     DULP              ;BR IF DRIVE NOT AVAILABLE
4676  024526  004037  031012                  JSR     R0,RD.RM          ;READ THE DRIVE TYPE REG.
4677  024532  000026                          RMDT
4678  024534  024732                          8$                        ;ERROR RETURN ADDRESS
4679  024536  012605                          MOV     (SP)+,R5          ;PUT DRIVE TYPE IN R5
4680  024540  112761  000004  024076          MOVB    #4,DRVTYP(R1)     ;SET RM03 INDICATOR
4681  024546  022705  020024                  CMP     #20024,R5         ;SINGLE PORT RM03 ?
4682  024552  001407                          BEQ     2$                ;BR IF YES
4683  024554  022705  024024                  CMP     #24024,R5         ;DUAL PORT RM03 ?
4684  024560  001404                          BEQ     2$                ;BR IF YES
4685  024562  112761  177777  024076          MOVB    #-1,DRVTYP(R1)    ;SET INDICATOR TO 'OTHER'
4686  024570  000446                          BR      6$                ;EXIT
4687  024572  012746  000121          2$:     MOV     #121,-(SP)        ;DO A "READ-IN PRESET"
4688  024576  004037  031172                  JSR     R0,WRT.RM
4689  024602  000000                          RMCS1
4690  024604  024732                          8$
4691  024606  012746  010000                  MOV     #BIT12,-(SP)      ;SET FMT22=1
4692  024612  004037  031172                  JSR     R0,WRT.RM
4693  024616  000032                          RMOF
4694  024620  024732                          8$
4695  024622  004037  031012                  JSR     R0,RD.RM          ;READ RMDS
4696  024626  000012                          RMDS
4697  024630  024732                          8$
4698  024632  012605                          MOV     (SP)+,R5          ;AND SAVE IT IN R5
4699  024634  100015                          BPL     4$                ;BRANCH IF ATA=0
4700  024636  116164  024202  000016          MOVB    ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
4701  024644  004037  031012                  JSR     R0,RD.RM          ;FIND OUT WHY ATA=1
4702  024650  000014                          RMER1
4703  024652  024732                          8$
4704  024654  006126                          ROL     (SP)+             ;IS IT UNSAFE?
4705  024656  100004                          BPL     4$                ;BR IF NOT
4706  024660  112761  177777  024066          MOVB    #-1,DRVSTA(R1)    ;SET UNSAFE INDICATOR
4707  024666  000407                          BR      6$                ;EXIT
4708  024670  005105                  4$:     COM     R5                ;CHECK MOL, DPR, DRY, AND VV
4709  024672  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
4710  024676  001003                          BNE     6$                ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
4711  024700  112761  000001  024066          MOVB    #1,DRVSTA(R1)     ;SET DRIVE STATUS TO ONLINE
4712  024706  005720                  6$:     TST     (R0)+             ;STEP OVER THE ERROR RETURN
4713  024710  000410                          BR      8$                ;EXIT
4714  024712  006301                  7$:     ASL     R1                ;CHANGE INDEX TO ADDRESS WORDS
4715  024714  012761  003720  024160          MOV     #2000.,TIMER(R1)  ;START 2 SEC TIMER
4716  024722  006201                          ASR     R1                ;RESTORE R1
4717  024724  112761  177777  024106          MOVB    #-1,DPINT(R1)     ;SET PORT INITIALIZE INIDICATOR
4718  024732  012605                  8$:     MOV     (SP)+,R5          ;RESTORE R5
4719  024734  000200                          RTS     R0                ;EXIT
4720
4721                                  ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
4722                                  ;
4723                                  ;CALL
4724                                  ;
4725                                  ;       JSR     R0,@#RM03         ;CALL THE RM03 DRIVER
4726                                  ;       PNTADR                    ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
4727                                  ;       RETURN1                   ;RETURN HERE IF QUEUE IS FULL
4728                                  ;       RETURN2                   ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
4729                                                                    ;IS AN ERROR CONDITION
```

```
4730
4731  024736  013746  177776          RM03:   MOV     @#PS,-(SP)      ;SAVE THE CALLING STATUS
4732  024742  013737  024220  177776          MOV     RMVEC+2,@#PS    ;DON'T ALLOW ANY RM03 INTERRUPTS
4733  024750  112737  000001  024132          MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
4734  024756  104412                          SAVREG                  ;SAVE R0 - R5
4735  024760  011002                          MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
4736  024762  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
4737  024766  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
4738  024770  013704  024214                  MOV     RMADR,R4        ;UNIBUS ADDRESS OF RMCS1
4739  024774  105761  024066                  TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
4740  025000  003014                          BGT     1$              ;BRANCH IF ONLINE
4741  025002  105761  024134                  TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
4742  025006  001036                          BNE     3$              ;BRANCH IF YES
4743  025010  105761  024106                  TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
4744  025014  001042                          BNE     5$              ;BR IF YES
4745  025016  004037  024444                  JSR     R0,DRVINT       ;GO INIT. THE DRIVE
4746  025022  000434                          BR      4$              ;ERROR RETURN
4747  025024  105761  024066                  TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
4748  025030  003445                          BLE     6$              ;BR IF NOT
4749  025032  105761  024116          1$:     TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
4750  025036  001031                          BNE     5$              ;BR IF YES
4751  025040  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
4752  025044  004037  032144                  JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
4753  025050  000460                          BR      9$              ;QUEUE IS FULL
4754  025052  122762  000103  000002          CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
4755  025060  001003                          BNE     2$              ;BR IF NO
4756  025062  112761  177777  024134          MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
4757  025070  105761  024056          2$:     TSTB    DRVACT(R1)      ;IS THIS DRIVE ACTIVE?
4758  025074  001043                          BNE     8$              ;BR IF YES
4759  025076  004737  025230                  JSR     PC,OPT          ;CALL THE OPTIMIZER
4760  025102  000440                          BR      8$
4761  025104  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
4762  025112  000434                          BR      8$              ;EXIT
4763  025114  004737  026310          4$:     JSR     PC,CI7          ;GO HANDLE THE PARITY ERROR
4764  025120  000431                          BR      8$
4765  025122  004037  032144          5$:     JSR     R0,DRVQUE       ;PUT REQUEST IN QUEUE
4766  025126  000431                          BR      9$              ;QUEUE IS FULL
4767  025130  032714  000100                  BIT     #BIT06,(R4)     ;IE BIT SET ?
4768  025134  001023                          BNE     8$              ;YES
4769  025136  004737  031502                  JSR     PC,SET.IE       ;SET THE INTERRUPT
4770  025142  000420                          BR      8$              ;RETURN
4771  025144  105761  024066          6$:     TSTB    DRVSTA(R1)      ;SEE IF DRIVE OFFLINE OR UNSAFE
4772  025150  002412                          BLT     7$              ;BR IF UNSAFE
4773  025152  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE ERROR INDICATOR
4774  025160  105761  024076                  TSTB    DRVTYP(R1)      ;SEE IF OFFLINE OR NONEXISTENT
4775  025164  001007                          BNE     8$              ;BR IF OFFLINE
4776  025166  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;REPORT DRIVE NONEXISTENT
4777  025174  000403                          BR      8$              ;GO TO EXIT
4778  025176  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2)  ;DRIVE IS UNSAFE
4779  025204  104413                  8$:     RESREG                  ;RESTORE R0 - R5
4780  025206  005720                          TST     (R0)+           ;SETUP FOR NORMAL RETURN
4781  025210  000401                          BR      10$             ;FINISH UP, THEN EXIT
4782  025212  104413                  9$:     RESREG                  ;RESTORE R0 - R5
4783  025214  005720                  10$:    TST     (R0)+           ;CORRECT THE RETURN ADDRESS
4784  025216  105037  024132                  CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
4785  025222  012637  177776                  MOV     (SP)+,@#PS      ;RETURN "PS" TO USER LEVEL
```

```
                                        RTS     R0              ;RETURN TO CALLER
                                ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                ;       JSR     PC,OPT          ;SETUP A COMMAND
                                ;
025230  104412                  OPT:    SAVREG                  ;SAVE R0 - R5
025232  013746  177776                  MOV     @#PS,-(SP)      ;SAVE PROC. STATUS
025236  146137  024202  024130          BICB    ATABIT(R1),SRCHWT       ;CLEAR LA SEACH FLAG
025244  105061  024116                  CLRB    DPRQS(R1)       ;RESET THE PORT REQ FLAG ****
025250  004737  032220                  JSR     PC,GETREQ       ;GET "DPB" POINTER OF REQUEST
025254  005702                          TST     R2              ;IS THERE A REQUEST IN QUEUE?
025256  001472                          BEQ     7$              ;NO--BRANCH TO EXIT
025260  010164  000010                  MOV     R1,RMCS2(R4)    ;LOAD THE DRIVE ADDRESS *******
025264  012764  000111  000000          MOV     #111,RMCS1(R4)  ;CLEAR THE DRIVE
025272  032764  004000  000000          BIT     #BIT11,RMCS1(R4)        ;DVA SET ?
025300  001446                          BEQ     5$              ;TO PROT REQUEST ,IF NOT
025302  105761  024066          10$:    TSTB    DRVSTA(R1)      ;IS DRIVE ONLINE?
025306  003014                          BGT     1$              ;YES--BRANCH
025310  004737  032242                  JSR     PC,POPQUE       ;NO--REMOVE REQUEST FROM QUEUE
025314  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE STATUS/ERROR INDICATOR
025322  105761  024066                  TSTB    DRVSTA(R1)      ;IS DRIVE UNSAFE ?
025326  100053                          BPL     8$              ;BR TO EXIT IF NOT
025330  012762  110000  000016          MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
025336  000447                          BR      8$              ;BRANCH TO EXIT
025340                          1$:
                                ;       MOV     #111,-(SP)      ;LOAD COMMAND ONTO THE STACK
                                ;       JSR     R0,WRT.RM       ;LOAD THE REGISTER
                                ;       RMCS1                   ;REGISTER INCREMENT
                                ;       6$                      ;ERROR RETURN ADDRESS
                                ;       BIT     #BIT11,(R4)     ;DRIVE AVAILABLE ?
                                ;       BEQ     9$              ;BR IF NOT
025340  122762  000150  000002          CMPB    #150,2(R2)      ;IS THE REQUEST FOR I/O?
025346  002403                          BLT     2$              ;YES--BRANCH
025350  004737  025674                  JSR     PC,CI4          ;CALL THE COMMAND INITIATOR
025354  000440                          BR      8$              ;BRANCH TO EXIT
025356  005737  024200          2$:     TST     DTUW            ;DATA TRANSFER UNDERWAY?
025362  002012                          BGE     4$              ;YES--GO START A SEARCH
025364  005737  024156                  TST     SEEKFG          ;DO IMPLIED SEEKS?
025370  100404                          BMI     3$              ;YES---BRANCH
025372  004037  026644                  JSR     R0,LA           ;NO--DO LOOK AHEAD
025376  000427                          BR      8$              ;RETURN HERE ON A PARITY ERROR
025400  000403                          BR      4$              ;GO START A SEARCH
025402  004737  025466          3$:     JSR     PC,CI1          ;START A DATA TRANSFER
025406  000423                          BR      8$
025410  004737  025574          4$:     JSR     PC,CI3          ;START A SEARCH
025414  000420                          BR      8$              ;GO TO THE EXIT
025416  112761  177777  024116  5$:     MOVB    #-1,DPRQS(R1)   ;SET PORT REQUEST INDICATOR
025424  010103                          MOV     R1,R3           ;SET UP TO ADDRESS WORDS
025426  006303                          ASL     R3              ;CONVERT TO WORD INDEX
025430  012763  023420  024160          MOV     #10000.,TIMER(R3)  ;START 10 SEC TIMER
025436  000402                          BR      7$              ;EXIT
025440  004737  026310          6$:     JSR     PC,CI7          ;PROCESS THE PARITY ERROR
025444  032714  000100          7$:     BIT     #BIT06,(R4)     ;SEE IF 'IE' ALREADY SET
```

```
                                        BNE     8$              ;BR IF SET
                                        JSR     PC,SET.IE       ;SET "IE" WITHOUT A "TRE"
                                8$:     MOV     (SP)+,@#PS      ;RESTORE PROC. STATUS
                                        RESREG                  ;RESTORE R0 - R5
                                        RTS     PC

                                ;COMMAND INITIATOR
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                ;       MOV     #DPB,R2         ;ADDRESS OF DPB
                                ;       JSR     PC,CI?          ;CI?= CI1,CI3, OR CI4
                                ;                               ;WHERE:
                                ;                               ;CI1=DATA TRANSFER
                                ;                               ;CI2=SEARCH REQUESTED BY DATA XFER
                                ;                               ;CI4=NOT DATA TRANSFER

                                CI1:    JSR     PC,POPQUE       ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
                                        MOV     R2,TRNSWT       ;PUT REQ. IN TRANSFER WAIT QUEUE
                                        MOV     R2,R3           ;DPB ADDRESS TO R3
                                        MOV     RMADR,R4        ;RMCS1 ADDRESS
                                        MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                        ADD     #4,R3           ;DESIRED WORD COUNT
                                        ADD     #2,R4           ;RMWC ADDRESS
                                        MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
                                        MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
                                        MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
                                        JSR     R0,WRT.RM       ;CALL THE LOAD(WRITE) ROUTINE
                                        RMDA                    ;INDEX OF REGISTER TO LOAD
                                        CI7                     ;ERROR RETURN ADDRESS
                                        MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
                                        JSR     R0,WRT.RM
                                        RMDC
                                        CI7
                                        MOV     2(R2),-(SP)     ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
                                        JSR     R0,WRT.RM
                                        RMCS1
                                        CI7
                                        MOV     R1,DTUW         ;SET "DATA TRANSFER UNDERWAY"
                                        JMP     CI5
                                CI3:    MOV     RMADR,R4        ;RMCS1 ADDRESS
                                        MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                        MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
                                        JSR     R0,WRT.RM
                                        RMDC
                                        CI7
                                        MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
                                        SUB     MXWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
                                        BGE     1$
                                        ADD     #32.,R3
                                1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
                                        BIC     #177740,(SP)    ;CHOP OF ALL EXENTED BITS ,IF ANY
                                        MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
                                        JSR     R0,WRT.RM       ;LOAD DESIRED TRACK & SECTOR
                                        RMDA
                                        CI7
```

```
 4898  025650  012746  000131                        MOV     #131,-(SP)      ;START A SEARCH
 4899  025654  004037  031172                        JSR     R0,WRT.RM
 4900  025660  000000                                RMCS1
 4901  025662  026310                                CI7
 4902  025664  156137  024202  024130                BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
 4903  025672  000567                                BR      CI5
 4904  025674  013704  024214              CI4:      MOV     RMADR,R4        ;RMCS1 ADDRESS
 4905  025700  010164  000010                        MOV     R1,RMCS2(R4)    ;SELECT DRIVE
 4906  025704  116203  000002                        MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
 4907  025710  122703  000131                        CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
 4908  025714  001007                                BNE     1$              ;BRANCH IF NO
 4909  025716  016246  000010                        MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
 4910  025722  004037  031172                        JSR     R0,WRT.RM
 4911  025726  000006                                RMDA
 4912  025730  026310                                CI7
 4913  025732  000403                                BR      2$              ;GO LOAD CYLINDER
 4914  025734  122703  000105              1$:       CMPB    #105,R3         ;IS IT A SEEK COMMA
 4915  025740  001007                                BNE     3$              ;BRANCH IF NO
 4916  025742  016246  000012              2$:       MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
 4917  025746  004037  031172                        JSR     R0,WRT.RM
 4918  025752  000034                                RMDC
 4919  025754  026310                                CI7
 4920  025756  000546                                BR      CI6
 4921  025760  122703  000115              3$:       CMPB    #115,R3         ;IS IT AN "OFFSET" COMMAND?
 4922  025764  001013                                BNE     4$              ;BR IF NO
 4923  025766  004037  031012                        JSR     R0,RD.RM        ;MERGE THE OFFSET VALUE INTO RMOF
 4924  025772  000032                                RMOF                    ;BUT DON'T CHANGE THE UPPER
 4925  025774  026310                                CI7
 4926  025776  116216  000001                        MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
 4927  026002  004037  031172                        JSR     R0,WRT.RM       ;REGISTER (RMOF)
 4928  026006  000032                                RMOF
 4929  026010  026310                                CI7
 4930  026012  000530                                BR      CI6             ;GO START THE COMMAND
 4931  026014  122703  000107              4$:       CMPB    #107,R3         ;IS IT A "RECALIBRATE" COMMAND?
 4932  026020  001525                                BEQ     CI6             ;BRANCH IF YES
 4933  026022  122703  000117                        CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
 4934  026026  001522                                BEQ     CI6             ;BRANCH IF YES
 4935  026030  122703  000103                        CMPB    #103,R3         ;IS IT AN "UNLOAD" COMMAND?
 4936  026034  001016                                BNE     5$              ;BRANCH IF NO
 4937  026036  112761  000001  024056                MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
 4938  026044  105061  024066                        CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
 4939  026050  112761  000001  024134                MOVB    #1,ULDFLG(R1)   ;SET "UNLOAD IN PROGRESS" FLAG
 4940  026056  010346                                MOV     R3,-(SP)        ;START THE "UNLOAD" COMMAND
 4941  026060  004037  031172                        JSR     R0,WRT.RM
 4942  026064  000000                                RMCS1
 4943  026066  026310                                CI7
 4944  026070  000207                                RTS     PC              ;RETURN TO USER
 4945  026072  122703  000143              5$:       CMPB    #143,R3         ;IS IT A "SET FORMAT" COMMAND?
 4946  026076  001014                                BNE     6$              ;BRANCH IF NO
 4947  026100  004037  031012                        JSR     R0,RD.RM        ;READ THE OFFSET REGISTER
 4948  026104  000032                                RMOF
 4949  026106  026310                                CI7
 4950  026110  116266  000001  000001                MOVB    1(R2),1(SP)     ;COMBINE "FMT22","ECI", AND "HCI"
 4951  026116  004037  031172                        JSR     R0,WRT.RM       ;LOAD "FMT22", "ECI", AND/OR "HCI".
 4952  026122  000032                                RMOF
 4953  026124  026310                                CI7
```

```
4954  026126  000436                        BR      12$
4955  026130  122703  000141                6$:     CMPB    #141,R3         ;IS IT A "GET REGISTER" COMMAND?
4956  026134  001023                        BNE     10$                     ;BRANCH IF NO
4957  026136  016203  000006                7$:     MOV     6(R2),R3        ;POINTS TO 1ST ADDRESS OF WHERE
4958                                                                        ;TO PUT THE REGISTER(S)
4959  026142  116237  000010  026160                MOVB    10(R2),9$       ;INIT. THE INDEX FOR THE FIRST REG.
4960  026150  116205  000011                      MOVB    11(R2),R5       ;INDEX OF LAST REG. TO MOVE
4961  026154  004037  031012                8$:     JSR     R0,RD.RM        ;READ RH70/RM03 REGISTER
4962  026160  000000                        9$:     RMCS1                   ;INDEX OF REG. TO READ
4963  026162  026310                                CI7
4964  026164  012623                                MOV     (SP)+,(R3)+     ;GET THE CONTENTS OF RH70//RM03 REG.
4965  026166  023705  026160                        CMP     9$,R5           ;LAST REG. BEEN READ?
4966  026172  001414                                BEQ     12$             ;GET OUT IF YES
4967  026174  062737  000002  026160                ADD     #2,9$           ;INCREASE THE INDEX BY 2
4968  026202  000764                                BR      8$              ;LOOP--MORE TO READ
4969  026204  122703  000145                10$:    CMPB    #145,R3         ;IS IT A "SELECT DRIVE" COMMAND?
4970  026210  001405                                BEQ     12$             ;BRANCH IF YES
4971  026212  010346                        11$:    MOV     R3,-(SP)        ;LOAD THE COMMAND
4972  026214  004037  031172                        JSR     R0,WRT.RM
4973  026220  000000                                RMCS1
4974  026222  026310                                CI7
4975  026224  004737  032242                12$:    JSR     PC,POPQUE       ;REMOVE REQ. FROM QUEUE
4976  026230  052762  000200  000016                BIS     #BIT07,16(R2)   ;SET THE "DONE" BIT
4977  026236  005737  024154                        TST     SAVEFG          ;SAVE THE RH70/RM03 REGISTERS?
4978  026242  100002                                BPL     13$             ;BRANCH IF NO
4979  026244  004737  031364                        JSR     PC,SVRH70       ;YES--GO SAVE THE REGISTERS
4980  026250  000207                        13$:    RTS     PC              ;RETURN TO USER
4981  026252  006301                        CI5:    ASL     R1
4982  026254  012761  001750  024160                MOV     #1000.,TIMER(R1)        ;SET A ONE SECOND TIMER
4983  026262  006201                                ASR     R1
4984  026264  112761  000001  024056                MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE
4985  026272  000207                                RTS     PC              ;RETURN TO THE USER
4986  026274  010346                        CI6:    MOV     R3,-(SP)        ;LOAD THE COMMAND
4987  026276  004037  031172                        JSR     R0,WRT.RM
4988  026302  000000                                RMCS1
4989  026304  026310                                CI7
4990  026306  000761                                BR      CI5
4991  026310  032764  010000  000010        CI7:    BIT     #BIT12,RMCS2(R4)  ;DRIVE NON-EXISTENT ?
4992  026316  001034                                BNE     CI8             ;BR IF YES
4993  026320  005702                        1$:     TST     R2              ;ANYTHING IN QUEUE ?
4994  026322  001405                                BEQ     CI7B            ;BR IF NOT
4995  026324  012762  104000  000016                MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
4996  026332  004737  031364                        JSR     PC,SVRH70       ;GO SAVE THE RH70/RM03 REGISTERS
4997  026336  012746  000111                CI7B:   MOV     #111,-(SP)      ;DO A "DRIVE CLEAR"
4998  026342  004037  031172                        JSR     R0,WRT.RM
4999  026346  000000                                RMCS1
5000  026350  026410                                CI8
5001  026352  004737  032124                        JSR     PC,EMPTYQ       ;EMPTY THE QUEUE
5002  026356  105061  024134                        CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD IN QUEUE FLAG
5003  026362  105061  024056                        CLRB    DRVACT(R1)      ;DRIVE IS IDLE
5004  026366  020137  024200                        CMP     R1,DTUW         ;IF THIS DRIVE HAD AN I/O REQUEST
5005  026372  001005                                BNE     1$              ;IN PROGRESS CLEAR ALL OF THE FLAGS
5006  026374  005037  024126                        CLR     TRNSWT
5007  026400  012737  177777  024200                MOV     #-1,DTUW
5008  026406  000207                        1$:     RTS     PC
5009  026410  104412                        CI8:    SAVREG                  ;SAVE R0 - R5
```

```
5010  026412  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;IS 'NED' SET ?
5011  026420  001002                          BNE     1$                ;BR IF YES
5012  026422  005001                          CLR     R1
5013  026424  005003                          CLR     R3
5014  026426  105761  024056            1$:   TSTB    DRVACT(R1)        ;DRIVE ACTIVE?
5015  026432  001443                          BEQ     5$                ;BRANCH IF NO
5016  026434  013702  024126                  MOV     TRNSWT,R2         ;GET THE "TRANSFER WAIT" QUEUE
5017  026440  020137  024200                  CMP     R1,DTUW           ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
5018  026444  001402                          BEQ     2$                ;BRANCH IF YES
5019  026446  004737  032220                  JSR     PC,GETREQ         ;GET THE DPB POINTER
5020  026452  005702                    2$:   TST     R2                ;QUEUE ENTRY FOR DRIVE ?
5021  026454  001415                          BEQ     4$                ;BR IF NOT
5022  026456  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
5023  026464  001404                          BEQ     3$                ;BR IF NOT
5024  026466  012762  100002  000016          MOV     #BIT15!BIT01,16(R2) ;SET 'DRIVE NON-EXISTENT' INDICATOR
5025  026474  000405                          BR      4$                ;CONTINUE
5026  026476  012762  102000  000016    3$:   MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
5027  026504  004737  031364                  JSR     PC,SVRH70         ;SAVE RH70/RM03 REGISTERS
5028  026510  012763  177777  024160    4$:   MOV     #-1,TIMER(R3)     ;STOP THE TIMER
5029  026516  105061  024056                  CLRB    DRVACT(R1)        ;SET "DRIVE ACTIVE" TO IDLE
5030  026522  020137  024200                  CMP     R1,DTUW           ;IS THIS DRIVE SETUP FOR A TRANSFER
5031  026526  001005                          BNE     5$                ;BR IF NOT
5032  026530  012737  177777  024200          MOV     #-1,DTUW          ;RESET THE INDICATOR
5033  026536  005037  024126                  CLR     TRNSWT            ;CLEAR THE TRANSFER QUEUE
5034  026542  105061  024134            5$:   CLRB    ULDFLG(R1)        ;CLEAR UNLOAD FLAG
5035  026546  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
5036  026554  001021                          BNE     6$                ;BR IF YES
5037  026556  005201                          INC     R1                ;MOVE TO THE NEXT DRIVE
5038  026560  062703  000002                  ADD     #2,R3
5039  026564  042701  177770                  BIC     #^C7,R1
5040  026570  001316                          BNE     1$                ;BRANCH IF MORE DRIVES
5041  026572  012737  177777  024200          MOV     #-1,DTUW          ;NO DATA TRANSFERS UNDERWAY
5042  026600  005037  024126                  CLR     TRNSWT            ;CLEAR THE 'TRANSFER WAIT' QUEUE
5043  026604  004737  032046                  JSR     PC,CLRQUE         ;CLEAR ALL OF THE REQUEST QUEUES
5044  026610  012764  000040  000010          MOV     #BIT05,RMCS2(R4)  ;DO A MASSBUS INIT.
5045  026616  000406                          BR      7$                ;CONTINUE
5046  026620  004737  032124            6$:   JSR     PC,EMPTYQ         ;CLEAR THE DRIVE'S QUEUE
5047  026624  105061  024066                  CLRB    DRVSTA(R1)        ;SET DRIVE TO OFFLINE
5048  026630  105061  024076                  CLRB    DRVTYP(R1)        ;CLEAR THE DRIVE TYPE INDICATOR
5049  026634  004737  031502            7$:   JSR     PC,SET.IE         ;SET "IE" WITHOUT "TRE"
5050  026640  104413                          RESREG                    ;RESTORE R0 - R5
5051  026642  000207                          RTS     PC                ;RETURN
5052
5053                                  ;LOOK AHEAD ROUTINE
5054                                  ;
5055                                  ;CALL
5056                                  ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER
5057                                  ;       MOV     #DPB,R2           ;POINT TO DPB
5058                                  ;       JSR     R0,LA             ;GO CHECK THE WINDOW
5059                                  ;       RETURN1                   ;ERROR RETURN
5060                                  ;       RETURN2                   ;START A SEARCH
5061                                  ;       RETURN3                   ;START A DATA TRANSFER
5062
5063  026644  013704  024214    LA:   MOV     RMADR,R4          ;GET RMCS1'S ADDRESS
5064  026650  010164  000010          MOV     R1,RMCS2(R4)      ;SELECT DRIVE
5065  026654  004037  031012          JSR     R0,RD.RM          ;READ DRIVE STATUS
```

```
                                        RMDS
                                        4$
                                                        ;ERROR RETURN ADDRESS
                        BIC     #↑C020200,(SP) ;ON CYLINDER ?
                        CMP     #200,(SP)+      ;PIP=0,DRY=1?
                        BNE     3$              ;NO
                        INCB    LACNT(R1)       ;INCREMENT THE LOOK AHEAD COUNT
                        CMPB    LACNT(R1),MXLACT;EXCEED MAX?
                        BGT     2$              ;BRANCH IF YES
                        MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
                        SWAB    R3              ;MULT. BY 64--ALIGN WITH
                        ASR     R3              ;LOOK AHEAD REGISTER
                        ASR     R3
                        MOV     #340,@#PS       ;PRIORITY LEVEL "7"
                6$:     JSR     R0,RD.RM        ;READ LOOK AHEAD REGISTER
                        RMLA
                        4$
                        CMP     (SP),RMLA(R4)   ;CORRECT LA NUMBER ?
                        BEQ     7$              ;YES
                        TST     (SP)+           ;NO,CLEAR STACK
                        BR      3$
                7$:     SUB     (SP)+,R3        ;CALCULATE THE DELTA
                        BGE     1$
                        ADD     #<32.*64.>,R3   ;MAKE THE DELTA POSITIVE
                1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
                        BLT     3$              ;IF IT IS WITHIN THE
                        CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
                        BGE     3$              ;THE LOOK AHEAD COUNT
                2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
                        TST     (R0)+
                3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
                        BR      5$              ;EXIT
                4$:     JSR     PC,CI7          ;PROCESS THE ERROR
                5$:     RTS     R0              ;RETURN

                ;INTERRUPT SERVICE ROUTINE

                ISR:    MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
                        SAVREG                  ;SAVE R0 - R5
                        MOV     RMADR,R4        ;ADDRESS OF RHSCS1
                        MOV     DTUW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
                        BLT     1$              ;BRANCH IF NO DATA TRANSFER UNDERWAY
                        JSR     PC,TD           ;CALL TRANSFER DONE
                        BR      2$              ;EXIT
                1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
                2$:     RESREG                  ;RESTORE R0 - R5
                        CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
                        RTI                     ;RETURN

                ;TRANSFER DONE ROUTINE

                TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
                        MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
                        ASL     R1
                        MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
                        ASR     R1
                        MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
```

```
5122  027114  005037  024126                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
5123  027120  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET DONE
5124  027126  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
5125  027132  004037  031012                  JSR     R0,RD.RM        ;TRANSFER ERROR(TRE=1)?
5126  027136  000000                          RMCS1
5127  027140  026310                          CI7
5128  027142  006126                          ROL     (SP)+
5129  027144  100417                          BMI     3$              ;BR IF YES
5130  027146  005737  024154                  TST     SAVEFG          ;SAVE THE RH70/RM03 REGISTERS?
5131  027152  100002                          BPL     1$              ;BRANCH IF NO
5132  027154  004737  031364                  JSR     PC,SVRH70       ;YES--SAVE THE REGISTERS
5133  027160                          1$:
5134  027160  004737  032220                  JSR     PC,GETREQ       ;GET DPB POINTER
5135  027164  005702                          TST     R2              ;ENTRY FOR DRIVE ?
5136  027166  001403                          BEQ     2$              ;BR IF NOT
5137  027170  004737  025230                  JSR     PC,OPT          ;CALL OPTIMIZER
5138  027174  000417                          BR      SC              ;CHECK OTHER DRIVES
5139                                  ;THE RELEASE DRIVE COMMAND IS FORECD TO ENTER FOR DUAL PORT OPERATION
5140  027176  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
5141  027202  000414                          BR      SC              ;CHECK FOR OTHER DRIVES
5142  027204  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
5143  027212  004737  032124                  JSR     PC,EMPTYQ       ;EMPTY THE "DRIVE'S WAIT" QUEUE
5144  027216  004737  031364                  JSR     PC,SVRH70       ;SAVE THE RH70/RM03 REGISTERS
5145  027222  012714  040111                  MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
5146  027226  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
5147  027232  000400                          BR      SC              ;CHECK FOR OTHER DRIVES
5148
5149
5150
5151                                  ;SPECIAL CONDITION ROUTINE
5152
5153  027234  116403  000016          SC:     MOVB    RMAS(R4),R3     ;READ "RMAS"
5154  027240  001014                          BNE     2$              ;BRANCH IF ANY 'ATA' BITS SET
5155  027242  004037  031012                  JSR     R0,RD.RM        ;READ CONTROL AND STATUS REGISTER
5156  027246  000000                          RMCS1
5157  027250  026410                          CI8
5158  027252  106126                          ROLB    (SP)+           ;IS "IE"=1?
5159  027254  100405                          BMI     1$              ;YES, NO DRIVES TO CHECK
5160  027256  004037  032310                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5161  027262  104001                          ERROR   1               ;REPORT AN ILLEGAL INTERRUPT
5162  027264  004737  031502                  JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
5163  027270  000207                  1$:     RTS     PC              ;RETURN
5164  027272  005046                  2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
5165  027274  110316                          MOVB    R3,(SP)         ;AN "ATA"=1
5166  027276  012703  000001                  MOV     #1,R3
5167  027302  005001                          CLR     R1
5168  027304  030316                  SC3:    BIT     R3,(SP)         ;ATA=1?
5169  027306  001005                          BNE     SC5             ;YES--BRANCH
5170  027310  005201                  SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
5171  027312  106303                          ASLB    R3
5172  027314  001373                          BNE     SC3             ;BRANCH IF MORE TO CHECK?
5173  027316  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
5174  027320  000207                          RTS     PC              ;RETURN TO USER
5175  027322  105761  024106          SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
5176  027326  001402                          BEQ     1$              ;BR IF NOT
5177  027330  000137  030246                  JMP     SC13            ;PROCESS THE DRIVE
```

```
5178  027334  105761  024116          1$:   TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
5179  027340  001402                        BEQ     2$              ;BR IF NOT
5180  027342  000137  030246                JMP     SC13            ;START THE OUTSTANDING COMMAND
5181  027346  105761  024066          2$:   TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
5182  027352  003025                        BGT     5$              ;BRANCH IF ONLINE
5183  027354  105761  024134                TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
5184  027360  003422                        BLE     5$              ;BRANCH IF NOT
5185  027362  004737  032220                JSR     PC,GETREQ       ;GET DPB POINTER
5186  027366  004737  031364                JSR     PC,SVRH70       ;SAVE THE RH70/RM03 REGISTERS
5187  027372  004737  030176                JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
5188                                                                ;ALSO DO A DRIVE INIT (DRVINT)
5189  027376  105761  024066                TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
5190  027402  003416                        BLE     6$              ;NO---BRANCH
5191  027404  032737  040000  024046        BIT     #BIT14,RMERRS   ;WAS THERE AN ERROR?
5192  027412  001002                        BNE     3$              ;BR IF ERROR
5193  027414  000137  030066                JMP     SC11            ;NO ERROR
5194  027420  013705  024050          3$:   MOV     RMERRS+2,R5     ;YES -- PICKUP RMER1 AND
5195  027424  000502                        BR      SC6A            ;GO PROCESS THE ERROR
5196  027426  105761  024056          5$:   TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
5197  027432  001033                        BNE     SC6             ;BR IF EITHER
5198  027434  004737  030176                JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
5199                                                                ;ALSO DO A DRVINT
5200  027440  105761  024106          6$:   TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
5201  027444  001321                        BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
5202  027446  105761  024066                TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
5203  027452  100412                        BMI     7$              ;BR IF UNSAFE
5204  027454  032737  020000  024052        BIT     #BIT13,RMERRS+4 ;ADDRESS PLUG CHANGED ?
5205  027462  001013                        BNE     8$              ;BR IF YES
5206  027464  012746  000113                MOV     #113,-(SP)      ;RELEASE COMMAND
5207  027470  004037  031172                JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RMCS1
5208  027474  000000                        RMCS1                   ;REGISTER INDEX
5209  027476  030036                        SC8                     ;PARITY EXIT ADDRESS
5210  027500  011605                7$:     MOV     (SP),R5         ;PICKUP (RMAS) BEFORE THE ERROR CALL
5211  027502  004037  032310                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5212  027506  104002                        ERROR   2               ;REPORT THE UNEXPECTED ATTENTION
5213  027510  000677                        BR      SC4             ;GO CHECK FOR MORE ATA'S
5214  027512                          8$:
5215  027512  004037  032310                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5216  027516  104005                        ERROR   5               ;REPORT THE ADDRESS PLUG CHANGE
5217  027520  000673                        BR      SC4             ;CHECK FOR MORE DRIVES
5218  027522  006301                SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
5219  027524  012761  177777  024160        MOV     #-1,TIMER(R1)   ;STOP THE TIMER
5220  027532  006201                        ASR     R1              ;RESTORE THE DRIVE ADDRESS
5221  027534  004737  032220                JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
5222  027540  010164  000010                MOV     R1,RMCS2(R4)    ;SELECT DRIVE
5223  027544  004037  031012                JSR     R0,RD.RM        ;READ THE RM03'S STATUS REG.
5224  027550  000012                        RMDS
5225  027552  030036                        SC8
5226  027554  011605                        MOV     (SP),R5         ;AND PUT IT IN R5
5227  027556  006126                        ROL     (SP)+           ;WAS THERE AN ERROR?
5228  027560  100407                        BMI     1$              ;BR IF ERROR
5229  027562  105761  024056                TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
5230  027566  003137                        BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
5231  027570  052762  100210  000016        BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
5232  027576  000470                        BR      SC7
5233  027600  004037  031012          1$:   JSR     R0,RD.RM        ;READ ERROR REGISTER #1
```

```
5234  027604  000014                        RMER1
5235  027606  030036                        SC8
5236  027610  012605                        MOV     (SP)+,R5          ;AND SAVE IT IN R5
5237  027612  004737  031364                JSR     PC,SVRH70         ;SAVE RH70/RM03 REGISTERS
5238  027616  012746  000111                MOV     #111,-(SP)        ;ISSUE A DRIVE CLEAR
5239  027622  004037  031172                JSR     R0,WRT.RM
5240  027626  000000                        RMCS1
5241  027630  030036                        SC8
5242  027632  006105              SC6A:     ROL     R5                ;WAS "UNSAFE" CONDITION =1?
5243  027634  100406                        BMI     1$                ;BRANCH IF YES
5244  027636  005702                        TST     R2                ;ANYTHING IN QUEUE ?
5245  027640  001447                        BEQ     SC7               ;BR IF NOT
5246  027642  052762  100240  000016        BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
5247  027650  000443                        BR      SC7
5248  027652  004037  031012      1$:       JSR     R0,RD.RM          ;READ DRIVE STATUS REG. #1
5249  027656  000012                        RMDS
5250  027660  030036                        SC8
5251  027662  011605                        MOV     (SP),R5           ;SAVE RMDS IN R5
5252  027664  006126                        ROL     (SP)+             ;"ERR"=1?
5253  027666  100011                        BPL     2$                ;BR IF NO--UNSAFE CLEARED
5254  027670  112761  177777  024066        MOVB    #-1,DRVSTA(R1)    ;DRIVE IS UNSAFE
5255  027676  004737  031364                JSR     PC,SVRH70         ;SAVE RH70/RM03 REGISTERS
5256  027702  052762  110000  000016        BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
5257  027710  000423                        BR      SC7
5258  027712  032705  010000      2$:       BIT     #BIT12,R5         ;"MOL" = 1 ?
5259  027716  001015                        BNE     3$                ;BR IF YES
5260  027720  112761  177777  024056        MOVB    #-1,DRVACT(R1)    ;ACTIVE ERROR RECOVER
5261  027726  112761  000001  024066        MOVB    #1,DRVSTA(R1)     ;ONLINE
5262  027734  006301                        ASL     R1
5263  027736  012761  072460  024160        MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
5264  027744  006201                        ASR     R1
5265  027746  000137  027310                JMP     SC4
5266  027752  052762  100220  000016  3$:   BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
5267  027760  105061  024056        SC7:    CLRB    DRVACT(R1)        ;DRIVE IS IDLE
5268  027764  004737  032124                JSR     PC,EMPTYQ         ;DUMP THE QUEUE
5269  027770  105761  024134                TSTB    ULDFLG(R1)        ;UNLOAD IN RMOGRESS OR QUEUE?
5270  027774  003002                        BGT     1$                ;BR IF NOT
5271  027776  105061  024134                CLRB    ULDFLG(R1)        ;CLEAR UNLOAD FLAG
5272  030002  116164  024202  000016  1$:   MOVB    ATABIT(R1),RMAS(R4)   ;CLEAR ATTENTION BIT
5273  030010  105761  024066                TSTB    DRVSTA(R1)        ;IS THE DRIVE UNSAFE ?
5274  030014  100406                        BMI     2$                ;BR IF IT IS
5275  030016  012746  000113                MOV     #113,-(SP)        ;RELEASE COMMAND
5276  030022  004037  031172                JSR     R0,WRT.RM         ;WRITE THE COMMAND INTO RPCS1
5277  030026  000000                        RMCS1                     ;REGISTER INDEX
5278  030030  030036                        SC8                       ;PARITY EXIT ADDRESS
5279  030032  000137  027310        2$:     JMP     SC4               ;CHECK FOR MORE DRIVES
5280  030036  105761  024056        SC8:    TSTB    DRVACT(R1)        ;IS DRIVE IDLE?
5281  030042  001405                        BEQ     1$                ;YES--BRANCH
5282  030044  004737  032220                JSR     PC,GETREQ         ;GET DPB POINTER
5283  030050  004737  026310                JSR     PC,CI7            ;PROCESS THE PARITY ERROR
5284  030054  000402                        BR      2$                ;CONTINUE
5285  030056  004737  026336        1$:     JSR     PC,CI7B           ;PROCESS THE UNCORRECTABLE PARITY ERROR
5286  030062  000137  027310        2$:     JMP     SC4               ;CHECK MORE DRIVES
5287  030066  105761  024134        SC11:   TSTB    ULDFLG(R1)        ;"UNLOAD IN PROGRESS"?
5288  030072  003402                        BLE     1$                ;BRANCH IF NO
5289  030074  105061  024134                CLRB    ULDFLG(R1)        ;CLEAR UNLOAD FLAG
```

```
5290  030100  105061  024056                1$:     CLRB    DRVACT(R1)          ;SET DRIVE IDLE
5291  030104  136137  024202  024130                BITB    ATABIT(R1),SRCHWT   ;DOING A SEARCH OPERATION FOR
5292                                                                              ;AN I/O COMMAND?
5293  030112  001012                                BNE     2$                  ;BRANCH IF YES
5294  030114  004737  032242                        JSR     PC,POPQUE           ;REMOVE REQUEST FROM QUEUE
5295  030120  052762  000200  000016                BIS     #BIT07,16(R2)       ;SET "DONE" BIT
5296  030126  005737  024154                        TST     SAVEFG              ;SAVE THE REGISTERS?
5297  030132  100002                                BPL     2$                  ;BRANCH IF NO
5298  030134  004737  031364                        JSR     PC,SVRH70           ;YES--SAVE ALL OF THE RH70/RM03 REG'S
5299  030140  116164  024202  000016    2$:         MOVB    ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
5300  030146  146137  024202  024130                BICB    ATABIT(R1),SRCHWT       ;CLEAR IMPLIED SEEK SET
5301  030154  006301                                ASL     R1                  ;WORD INDEX
5302  030156  012761  177777  024160                MOV     #-1,TIMER(R1)       ;STOP CLOCK
5303  030164  006201                                ASR     R1                  ;RESTORE R1
5304  030166  004737  025230                        JSR     PC,OPT              ;START A REQUEST
5305  030172  000137  027310                        JMP     SC4                 ;CHECK FOR MORE DRIVES
5306  030176  010164  000010            SC12:       MOV     R1,RMCS2(R4)        ;SELECT DRIVE
5307  030202  016437  000012  024046                MOV     RMDS(R4),RMERRS     ;SAVE THE FOUR REGISTERS THAT
5308  030210  016437  000014  024050                MOV     RMER1(R4),RMERRS+2      ;WILL TELL US SOMETHING
5309  030216  016437  000042  024052                MOV     RMER2(R4),RMERRS+4
5310  030224  016437  000040  024054                MOV     RMMR2(R4),RMERRS+6
5311  030232  004037  024444                        JSR     R0,DRVINT           ;INIT. THE STATE OF THE DRIVE
5312  030236  000401                                BR      1$                  ;TAKE ERROR EXIT
5313  030240  000207                                RTS     PC                  ;RETURN
5314  030242  005726                    1$:         TST     (SP)+               ;POP PC OFF OF THE STACK
5315  030244  000674                                BR      SC8                 ;PROCESS THE PARITY ERROR
5316  030246  006301                    SC13:       ASL     R1                  ;SETUP TO ADDRESS WORDS
5317  030250  012761  177777  024160                MOV     #-1,TIMER(R1)       ;STOP THE TIMER
5318  030256  006201                                ASR     R1                  ;
5319  030260  010164  000010                        MOV     R1,RMCS2(R4)        ;SELECT THE DRIVE
5320  030264  116164  024202  000016                MOVB    ATABIT(R1),RMAS(R4) ;CLEAR THE ATTENTION BIT
5321  030272  105761  024106                1$:     TSTB    DPINT(R1)           ;INITIALIZING THE DRIVE ?
5322  030276  001424                                BEQ     2$                  ;BR IF NOT
5323  030300  105061  024106                        CLRB    DPINT(R1)           ;CLEAR THE INIT INDICATOR
5324  030304  004037  024444                        JSR     R0,DRVINT           ;GO INIT THE DRIVE
5325  030310  000240                                NOP                         ;DUMMY PARITY ERROR RETURN
5326  030312  105761  024066                        TSTB    DRVSTA(R1)          ;DRIVE ONLINE ?
5327  030316  003014                                BGT     2$                  ;BR IF YES -- START ORDER
5328  030320  005702                                TST     R2                  ;QUEUE ENTRY FOR THE DRIVE
5329  030322  001426                                BEQ     3$                  ;BR IF NOT
5330  030324  004737  032220                        JSR     PC,GETREQ           ;GET DPB ADDRESS
5331  030330  052762  140000  000016                BIS     #BIT15!BIT14,16(R2)  ;INFORM USER THAT DRIVE OFFLINE
5332  030336  004737  031364                        JSR     PC,SVRH70           ;SAVE THE REGISTERS
5333  030342  004737  032124                        JSR     PC,EMPTYQ           ;EMPTY THE REQUEST QUEUE
5334  030346  000414                                BR      3$
5335  030350  032764  004000  000000    2$:         BIT     #BIT11,RMCS1(R4)            ;DVA SET ?
5336  030356  001006                                BNE     4$                  ;SET THEN CALL OPT
5337  030360  006301                                ASL     R1
5338  030362  012761  023420  024160                MOV     #10000.,TIMER(R1)
5339  030370  006201                                ASR     R1
5340  030372  000402                                BR      3$
5341  030374  004737  025230            4$:         JSR     PC,OPT              ;START THE PENDING REQUEST
5342  030400  000137  027310            3$:         JMP     SC4                 ;PROCESS OTHER DRIVES
5343
5344                                    ;/RM03 TIMER ROUTINE
5345                                    ;CALL
```

```
                                                ;       MOV     #TIME,-(SP)     ;ELASPED TIME IN MILLISECONDS ON THE STACK
                                                ;       JSR     PC,RPTMR        ;CALL RM03 TIME ROUTINE
030404  005737  024132                          RPTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
030410  001027                                          BNE     4$              ;IF NON ZERO EXIT
030412  112737  000001  024133                        MOVB    #1,ACTSTR       ;SET "ACTSTR"
030420  104412                                        SAVREG                  ;SAVE R0 - R5
030422  005001                                        CLR     R1              ;START WITH DRIVE 0
030424  005003                                        CLR     R3
030426  005763  024160                          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
030432  002406                                        BLT     2$              ;BRANCH IF NO
030434  166663  000002  024160                        SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
030442  003002                                        BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
030444  004737  030474                              JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
030450  005201                                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
030452  005723                                        TST     (R3)+
030454  022701  000010                              CMP     #8.,R1          ;OUT OF DRIVES?
030460  003362                                        BGT     1$              ;BRANCH IF NO
030462  104413                                  3$:     RESREG                  ;RESTORE R0 - R5
030464  105037  024133                              CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
030470  012616                                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
030472  000207                                        RTS     PC              ;RETURN

                                                ;SOFTWARE TIMEOUT ROUTINE
                                                ;
                                                ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
                                                ;       OR GREATER
                                                ;
                                                ;CALL:  STO
                                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                                ;       JSR     PC,STO          ;CALL
                                                ;       RETURN

030474  010146                                  STO:    MOV     R1,-(SP)        ;SAVE R1
030476  010246                                          MOV     R2,-(SP)        ;SAVE R2
030500  010346                                          MOV     R3,-(SP)        ;SAVE R3
030502  010446                                          MOV     R4,-(SP)        ;SAVE R4
030504  013704  024214                              MOV     RMADR,R4        ;GET ADDRESS OF "RMCS1"
030510  010164  000010                              MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
030514  004037  031012                              JSR     R0,RD.RM        ;READ "DRIVE STATUS REG"
030520  000012                                      RMDS
030522  030674                                      STO5
030524  105726                                      TSTB    (SP)+           ;IS "DRY"=1?
030526  100434                                      BMI     STO2            ;BR IF YES
030530  105761  024106                          STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
030534  001031                                      BNE     STO2            ;BR IF YES
030536  105761  024116                              TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
030542  001026                                      BNE     STO2            ;BR IF YES
030544  013702  024126                              MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
030550  020137  024200                              CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
030554  001404                                      BEQ     1$              ;BRANCH IF YES
030556  000137  031000                              JMP     STO9            ;IF NOT DON'T BOTHER DRIVES
030562  004737  032220                              JSR     PC,GETREQ       ;GET DPB ADDRESS
030566  052762  101000  000016  1$:                 BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
030574  004737  031364                              JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
030600  012764  000040  000010                      MOV     #BIT05,RMCS2(R4) ;"INIT" THE MASS BUS
```

```
5402  030606  105061  024056                  CLRB    DRVACT(R1)          ;DRIVE IS IDLE
5403  030612  105061  024134                  CLRB    ULDFLG(R1)          ;CLEAR THE UNLOAD FLAG
5404  030616  000470                          BR      ST09                ;DON'T BOTHER OTHER DRIVES
5405  030620  116405  000016          ST02:   MOVB    RMAS(R4),R5         ;READ ATTENTION REG
5406  030624  136105  024202                  BITB    ATABIT(R1),R5       ;IS ATTENTION FOR THIS DRIVE UP ?
5407  030630  001007                          BNE     ST03                ;YES--BRANCH
5408  030632  105761  024106                  TSTB    DPINT(R1)           ;TRYING TO INTIALIZE THE DRIVE ?
5409  030636  001021                          BNE     ST06                ;BR IF YES - DRIVE NOT ONLINE
5410  030640  105761  024116                  TSTB    DPRQS(R1)           ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
5411  030644  001035                          BNE     ST07                ;BR IF YES - NO RESPONSE TO REQUEST
5412  030646  000454                          BR      ST09                ;OTHER WISE EXIT
5413  030650  105761  024106          ST03:   TSTB    DPINT(R1)           ;INITIALIZING THE DRIVE ?
5414  030654  001003                          BNE     1$                  ;BR IF INIT PENDING
5415  030656  105761  024116                  TSTB    DPRQS(R1)           ;PORT REQUEST PENDING ?
5416  030662  001446                          BEQ     ST09                ;BR IF NOT
5417  030664  012763  177777  024160  1$:     MOV     #-1,TIMER(R3)       ;STOP THE TIMER
5418  030672  000442                          BR      ST09                ;EXIT
5419  030674  004737  026410          ST05:   JSR     PC,CI8              ;GO HANDLE THE PARITY ERROR
5420  030700  000437                          BR      ST09
5421  030702  105061  024106          ST06:   CLRB    DPINT(R1)           ;CLEAR THE INITIALIZE INDICATOR
5422  030706  105061  024066                  CLRB    DRVSTA(R1)          ;SET UNIT OFFLINE
5423  030712  012763  177777  024160          MOV     #-1,TIMER(R3)       ;STOP THE TIMER
5424  030720  004737  032220                  JSR     PC,GETREQ           ;GET THE DPB ADDRESS
5425  030724  005702                          TST     R2                  ;REQUEST IN QUEUE ?
5426  030726  001424                          BEQ     ST09                ;BR IF NOT
5427  030730  052762  140000  000016          BIS     #BIT15!BIT14,16(R2) ;INFORM THE USER DRIVE NOT AVAILABLE
5428  030736  000414                          BR      ST08                ;FINISH
5429  030740  012763  177777  024160  ST07:   MOV     #-1,TIMER(R3)       ;STOP THE TIMER
5430  030746  105061  024116                  CLRB    DPRQS(R1)           ;CLEAR PORT REQUEST INDICATOR
5431  030752  004737  032220                  JSR     PC,GETREQ           ;GET DPB ADDRESS
5432  030756  005702                          TST     R2                  ;QUEUE ENTRY FOR DRIVE ?
5433  030760  001407                          BEQ     ST09                ;BR IF NONE
5434  030762  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)  ;INFORM USER OF PORT REQUEST ERROR
5435  030770  004737  032124          ST08:   JSR     PC,EMPTYQ           ;CLEAR THE QUEUE FOR THE DRIVE
5436  030774  004737  031364                  JSR     PC,SVRH70           ;SAVE THE REGISTERS
5437  031000  012604                  ST09:   MOV     (SP)+,R4            ;RESTORE R4
5438  031002  012603                          MOV     (SP)+,R3            ;RESTORE R3
5439  031004  012602                          MOV     (SP)+,R2            ;RESTORE R2
5440  031006  012601                          MOV     (SP)+,R1            ;RESTORE R1
5441  031010  000207                          RTS     PC                  ;RETURN
5442
5443                                  ;ROUTINE TO READ A RH70/RM03 REGISTER
5444                                  ;
5445                                  ;CALL
5446                                  ;       JSR     R0,RD.RM            ;GO READ A REGISTER
5447                                  ;       INDEX                       ;REG. INDEX FROM BASE
5448                                  ;       ERRADR                      ;ERROR ADDRESS--PROCESS ERROR STARTING
5449                                  ;                                   ;AT THIS ADDRESS
5450                                  ;       RETURN                      ;CONTENTS OF REG. IS ON THE STACK
5451                                  ;
5452  031012  013737  024212  031160  RD.RM:   MOV     MCPEMX,RD.RM2       ;MAX. RETRYS ALLOWED
5453  031020  011646                          MOV     (SP),-(SP)          ;SAVE R0 FOR RETURN
5454  031022  013737  024214  031036          MOV     RMADR,RD.ADR        ;FORM THE DESIRED ADDRESS
5455  031030  062037  031036                  ADD     (R0)+,RD.ADR        ;USING THE BASE AND THE INDEX
5456  031034  013727                  RD.RM1: MOV     @(PC)+,(PC)+        ;READ THE DESIRED REGISTER OF THE RM03
5457  031036  000000                  RD.ADR: .WORD   0                   ;ADDRESS IS FORMED HERE
```

```
5458  031040  000000                  RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
5459  031042  013766  031040  000002          MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
5460  031050  013746  024214                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
5461  031054  062716  000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
5462  031060  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
5463  031064  001037                          BNE     RD.RM3          ;BR IF DRIVE NON-EXISTENT
5464  031066  017746  173122                  MOV     @RMADR,-(SP)    ;READ RMCS1
5465  031072  032716  020000                  BIT     #BIT13,(SP)     ;DID MCPE SET?
5466  031076  001002                          BNE     1$              ;BRANCH IF YES
5467  031100  022620                          CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
5468  031102  000432                          BR      RD.RM4          ;EXIT
5469  031104                          1$:
5470  031104  004037  032310                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN 'SESCAPE'
5471  031110  104003                          ERROR   3               ;REPORT "MCPE" ERROR
5472  031112  005737  024200                  TST     DTUW            ;DATA TRANSFER UNDERWAY?
5473  031116  100405                          BMI     2$              ;NO--BRANCH
5474  031120  032716  040000                  BIT     #BIT14,(SP)     ;NO--"TRE"=1?
5475  031124  001402                          BEQ     2$              ;NO--BRANCH
5476  031126  005726                          TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
5477  031130  000415                          BR      RD.RM3          ;TAKE THE FATAL ERROR EXIT
5478  031132  052716  040000          2$:     BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
5479  031136  000316                          SWAB    (SP)            ;POSITION BEFORE WRITING
5480  031140  013737  024214  031154          MOV     RMADR,3$        ;FORM ADDRESS OF HIGH BYTE
5481  031146  005237  031154                  INC     3$
5482  031152  112637                          MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RMCS1
5483  031154  000000                  3$:     .WORD   0               ;ADDRESS STORAGE
5484  031156  005327                          DEC     (PC)+           ;EXCEEDED MAX. RETRYS
5485  031160  000003                  RD.RM2: .WORD   3
5486  031162  002324                          BGE     RD.RM1          ;BRANCH IF NO
5487  031164  011000                  RD.RM3: MOV     (R0),R0         ;FATAL ERROR EXIT
5488  031166  012616                          MOV     (SP)+,(SP)
5489  031170  000200                  RD.RM4: RTS     R0
5490
5491                                  ;ROUTINE TO WRITE A REGISTER
5492                                  ;
5493                                  ;CALL
5494                                  ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
5495                                  ;       JSR     R0,WRT.RM       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
5496                                  ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
5497                                  ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
5498                                  ;       RETURN                  ;ERROR FREE RETURN
5499
5500  031172  013737  024212  031350  WRT.RM: MOV     MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
5501  031200  016637  000002  031260          MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
5502  031206  012616                          MOV     (SP)+,(SP)      ;ADJUST THE STACK
5503  031210  012037  031262                  MOV     (R0)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
5504  031214  001015                          BNE     1$              ;BRANCH IF NOT RMCS1
5505  031216  122737  000150  031260          CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
5506  031224  002411                          BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
5507  031226  004037  031012                  JSR     R0,RD.RM        ;NO---COMBINE A16&A17, & PSEL WITH
5508  031232  000000                          RMCS1                   ;THE COMMAND BEFORE SENDING IT TO
5509  031234  031354                          WRT.R3                  ;THE RH70/RM03
5510  031236  000316                          SWAB    (SP)
5511  031240  042716  177770                  BIC     #^C7,(SP)
5512  031244  112637  031261                  MOVB    (SP)+,WRT.WD+1
5513  031250  063737  024214  031262  1$:     ADD     RMADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
```

```
5514  031256  012737                  WRT.R1: MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
5515  031260  000000                  WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
5516  031262  000000                  WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
5517  031264  013746  024214                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
5518  031270  062716  000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
5519  031274  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
5520  031300  001025                          BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
5521  031302  004037  031012                  JSR     R0,RD.RM        ;CHECK FOR PARITY ERROR ON WRITE
5522  031306  000014                          RMER1
5523  031310  031354                          WRT.R3
5524  031312  032726  000010                  BIT     #BIT03,(SP)+
5525  031316  001420                          BEQ     WRT.R4          ;BRANCH IF "PAR=0"
5526  031320  016037  177776  031332          MOV     -2(R0),1$       ;PICKUP THE INDEX
5527  031326  004037  031012                  JSR     R0,RD.RM        ;READ THE REG.
5528  031332  000000                  1$:     .WORD   0               ;REG. INDEX
5529  031334  031354                          WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
5530  031336  004037  032310                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5531  031342  104004                          ERROR   4               ;REPORT THE PARITY ON WRITE ERROR
5532  031344  005726                          TST     (SP)+           ;CLEAR OFF THE STACK
5533  031346  005327                          DEC     (PC)+           ;DECREMENT THE ERROR COUNT
5534  031350  000003                  WRT.R2: .WORD   3               ;RETRY COUNTER
5535  031352  002341                          BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
5536  031354  011000                  WRT.R3: MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
5537  031356  000401                          BR      WRT.R5          ;EXIT
5538  031360  005720                  WRT.R4: TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
5539  031362  000200                  WRT.R5: RTS     R0
5540
5541                                  ;ROUTINE TO SAVE THE RH70/RP04/5/RM03 REGISTERS AS PER DPB+14
5542                                  ;
5543                                  ;CALL
5544                                  ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
5545                                  ;       JSR     PC,SVRH70       ;SAVE THE DRIVES REG'S
5546
5547  031364  104412                  SVRH70: SAVREG                  ;SAVE R0 - R5
5548  031366  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
5549  031370  001442                          BEQ     6$              ;BR IF NONE
5550  031372  013704  024214                  MOV     RMADR,R4
5551  031376  111264  000010                  MOVB    (R2),RMCS2(R4)  ;SELECT DRIVE
5552  031402  016203  000014                  MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
5553  031406  001433                          BEQ     6$              ;EXIT IF NO ADDRESS
5554  031410  005037  031444                  CLR     3$              ;COUNTER & POINTER
5555  031414  023727  031444  000022  1$:     CMP     3$,#RMDB        ;REACHED THE BUFFER REGISTER ?
5556  031422  001006                          BNE     2$              ;BR IF NOT
5557  031424  032764  000200  000010          BIT     #BIT07,RMCS2(R4)  ;'OR' SET ?
5558  031432  001002                          BNE     2$              ;BR IF SET
5559  031434  005023                          CLR     (R3)+           ;STORE RMDB AS ZEROES
5560  031436  000405                          BR      4$              ;CONTINUE
5561  031440  004037  031012          2$:     JSR     R0,RD.RM        ;READ THE SELECTED REGISTER
5562  031444  000000                  3$:     .WORD   0               ;REGISTER INDEX
5563  031446  031472                  5$                              ;ERROR RETURN ADDRESS
5564  031450  012623                          MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
5565  031452  023727  031444  000046  4$:     CMP     3$,#RMEC2       ;REACHED THE END ?
5566  031460  001406                          BEQ     6$              ;BR IF YES
5567  031462  062737  000002  031444          ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
5568  031470  000751                          BR      1$              ;CONTINUE READING THE REGISTERS
5569  031472  004737  026310          5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
```

```
031476  104413                  6$:     RESREG                  ;RESTORE R0 - R5
031500  000207                          RTS     PC              ;RETURN

                                ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                ;       JSR     PC,SET.IE       ;SET "IE"
                                ;       RETURN

031502  010446                  SET.IE: MOV     R4,-(SP)        ;SAVE R4
031504  013704  024214                  MOV     RMADR,R4        ;PICKUP ADDRESS OF RMCS1
031510  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
031514  011446                          MOV     (R4),-(SP)      ;READ RMCS1
031516  052716  040000                  BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
031522  000316                          SWAB    (SP)            ;ADJUST FOR DATO
031524  112714  000100                  MOVB    #BIT06,(R4)     ;SET "IE"
031530  032764  010000  000010          BIT     #BIT12,RMCS2(R4) ;IS "NED"=1?
031536  001002                          BNE     1$              ;YES--CLEAR "TRE"
031540  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
031542  000402                          BR      2$
031544  112664  000001          1$:     MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
031550  012604                  2$:     MOV     (SP)+,R4        ;RESTORE R4
031552  000207                          RTS     PC              ;RETURN TO CALLER

                                ;QUEUE COUNT
031554      000                 QCNT:   .BYTE   0               ;DRIVE 0
031555      000                         .BYTE   0               ;DRIVE 1
031556      000                         .BYTE   0               ;DRIVE 2
031557      000                         .BYTE   0               ;DRIVE 3
031560      000                         .BYTE   0               ;DRIVE 4
031561      000                         .BYTE   0               ;DRIVE 5
031562      000                         .BYTE   0               ;DRIVE 6
031563      000                         .BYTE   0               ;DRIVE 7

                                ;QUEUE INPUT POINTERS

031564  031646                  QINPT:  .WORD   QDRV0            ;DRIVE 0
031566  031666                          .WORD   QDRV1            ;DRIVE 1
031570  031706                          .WORD   QDRV2            ;DRIVE 2
031572  031726                          .WORD   QDRV3            ;DRIVE 3
031574  031746                          .WORD   QDRV4            ;DRIVE 4
031576  031766                          .WORD   QDRV5            ;DRIVE 5
031600  032006                          .WORD   QDRV6            ;DRIVE 6
031602  032026                          .WORD   QDRV7            ;DRIVE 7

                                ;QUEUE OUTPUT POINTERS

031604  031646                  QOUTPT: .WORD   QDRV0            ;DRIVE 0
031606  031666                          .WORD   QDRV1            ;DRIVE 1
031610  031706                          .WORD   QDRV2            ;DRIVE 2
031612  031726                          .WORD   QDRV3            ;DRIVE 3
031614  031746                          .WORD   QDRV4            ;DRIVE 4
031616  031766                          .WORD   QDRV5            ;DRIVE 5
031620  032006                          .WORD   QDRV6            ;DRIVE 6
031622  032026                          .WORD   QDRV7            ;DRIVE 7
```

```
 5626  031624  031646                QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
 5627  031626  031666                QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
 5628  031630  031706                        .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
 5629  031632  031726                        .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
 5630  031634  031746                        .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
 5631  031636  031766                        .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
 5632  031640  032006                        .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
 5633  031642  032026                        .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
 5634  031644  032046                        .WORD   QTERM           ;STOP DRIVE 7
 5635
 5636                                  ;DRIVE REQUEST QUEUES
 5637
 5638  031646  000010                QDRV0:  .BLKW   10
 5639  031666  000010                QDRV1:  .BLKW   10
 5640  031706  000010                QDRV2:  .BLKW   10
 5641  031726  000010                QDRV3:  .BLKW   10
 5642  031746  000010                QDRV4:  .BLKW   10
 5643  031766  000010                QDRV5:  .BLKW   10
 5644  032006  000010                QDRV6:  .BLKW   10
 5645  032026  000010                QDRV7:  .BLKW   10
 5646          032046                QTERM=.
 5647
 5648                                  ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
 5649                                  ;
 5650                                  ;CALL
 5651                                  ;       JSR     PC,CLRQUE
 5652
 5653  032046  104412                CLRQUE: SAVREG                  ;SAVE R0 - R5
 5654  032050  012702  031554                MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
 5655  032054  005022                        CLR     (R2)+           ;DRIVES 0 & 1
 5656  032056  005022                        CLR     (R2)+           ;DRIVES 2 & 3
 5657  032060  005022                        CLR     (R2)+           ;DRIVES 4 & 5
 5658  032062  005022                        CLR     (R2)+           ;DRIVES 6 & 7
 5659  032064  012703  000010                MOV     #8.,R3          ;MOVE THE STARTING
 5660  032070  012701  031624                MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
 5661  032074  012122                1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
 5662  032076  005303                        DEC     R3
 5663  032100  001375                        BNE     1$
 5664  032102  012703  000010                MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
 5665  032106  012701  031624                MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
 5666  032112  012122                2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
 5667  032114  005303                        DEC     R3
 5668  032116  001375                        BNE     2$
 5669  032120  104413                        RESREG                  ;RESTORE R0 - R5
 5670  032122  000207                        RTS     PC
 5671
 5672                                  ;EMPTY THE QUEUE SPECIFIED BY R1
 5673                                  ;
 5674                                  ;CALL
 5675                                  ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
 5676                                  ;       JSR     PC,EMPTYQ
 5677
 5678  032124  105061  031554       EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
 5679  032130  006301                        ASL     R1
 5680  032132  016161  031564  031604        MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
 5681  032140  006201                        ASR     R1
```

```
5682  032142  000207                          RTS     PC
5683
5684                                  ;ROUTINE TO PUT A REQUEST IN QUEUE
5685                                  ;
5686                                  ;CALL
5687                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
5688                                  ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
5689                                  ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
5690                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
5691                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
5692
5693  032144  122761  000010  031554  DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
5694  032152  001421                          BEQ     2$              ;BR IF YES-TAKE RETURN1
5695  032154  105261  031554                  INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
5696  032160  006301                          ASL     R1
5697  032162  010271  031564                  MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
5698  032166  062761  000002  031564          ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
5699  032174  026161  031564  031626          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
5700  032202  001003                          BNE     1$              ;BRANCH IF NO
5701  032204  016161  031624  031564          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
5702  032212  006201                  1$:     ASR     R1
5703  032214  005720                          TST     (R0)+           ;TAKE RETURN 2
5704  032216  000200                  2$:     RTS     R0              ;RETURN TO USER
5705
5706                                  ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
5707                                  ;
5708                                  ;CALL
5709                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
5710                                  ;       JSR     PC,GETREQ       ;GO GET THE REQUEST
5711                                  ;       RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
5712                                  ;                               ;R2=0 IF NO REQUEST IN QUEUE
5713
5714  032220  005002                  GETREQ: CLR     R2
5715  032222  105761  031554                  TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
5716  032226  001404                          BEQ     2$              ;NO---BRANCH
5717  032230  006301                  1$:     ASL     R1
5718  032232  017102  031604                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
5719  032236  006201                          ASR     R1
5720  032240  000207                  2$:     RTS     PC              ;RETURN TO USER
5721
5722                                  ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
5723                                  ;
5724                                  ;CALL
5725                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
5726                                  ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
5727                                  ;       RETURN                  ;R2=ADDRESS OF DPB REMOVED
5728
5729  032242  105361  031554          POPQUE: DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
5730  032246  006301                          ASL     R1
5731  032250  017102  031604                  MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
5732  032254  005071  031604                  CLR     @QOUTPT(R1)     ;REMOVE DPB ADDRESS FROM THE QUEUE
5733  032260  062761  000002  031604          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
5734  032266  026161  031604  031626          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
5735  032274  001003                          BNE     1$              ;NO--BRANCH TO EXIT
5736  032276  016161  031624  031604          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
5737  032304  006201                  1$:     ASR     R1
```

```
                                        RTS     PC              ;RETURN TO USER
                                ;ROUTINE TO SAVE THE CONTENTS OF 'SESCAPE' 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

032310  012037  032324          ES.SAV: MOV     (R0)+,1$        ;GET THE ERROR CALL
032314  013746  001174                  MOV     $ESCAPE,-(SP)   ;SAVE THE ADDRESS IN 'SESCAPE'
032320  005037  001174                  CLR     $ESCAPE         ;CLEAR THE ESCAPE RETURN
032324  000000                  1$:     .WORD   0               ;THE ERROR CALL IS MOVED HERE
032326  012637  001174                  MOV     (SP)+,$ESCAPE   ;RESTORE THE ESCAPE ADDRESS
032332  000200                          RTS     R0              ;RETURN


                                ;;*****************************************************************
                                .NLIST  BEX

                                .SBTTL  TELETYPE MESSAGES

032334  047440  043106  044514  UNTOFF: .ASCIZ  / OFFLINE/
032345  000040  047117  044514  UNTON:  .ASCIZ  / ONLINE/
032355  000040  047516  020124  NOTRM:  .ASCIZ  @ NOT AN RM03@
032372  047040  052117  050040  NOTPRS: .ASCIZ  / NOT PRESENT/
032407  000040  047125  040523  NOTSAF: .ASCIZ  / UNSAFE/
032417  000122  030120  000064  RM03B:  .ASCIZ  /RP04/
032424  050122  032460  000000  RP05:   .ASCIZ  /RP05/
032431  000122  030115  000063  RM03A:  .ASCIZ  /RM03/
032436  047125  052111  051440  SYSTAT: .ASCIZ  /UNIT STATUS/<CR><LF><LF>
032455  000015  042012  044522  MUNIT:  .ASCIZ  <CR><LF>/DRIVE: /
032467  000040  020057  000000  SLASH:  .ASCIZ  @ / @
032473  000103  000000          C:      .ASCIZ  /C/
032475  000124  000000          T:      .ASCIZ  /T/
032477  000060  000000          MDRVD:  .ASCIZ  /0/
032501  000040  000040          LIN4SP: .ASCII  /  /
032503  000040  000040          LINSP:  .ASCII  /  /
032506  047105  042524  020122  ENTADR: .ASCIZ  /ENTER ADDRESS LIMITS:/<CR><LF>
032536  020040  051104  053111  MOFFLN: .ASCIZ  /  DRIVE OFFLINE/<CR><LF>
032560  042040  044522  042526  MDRNP:  .ASCIZ  / DRIVE NOT PRESENT/<CR><LF>
032605  000040  051104  053111  MER11:  .ASCIZ  / DRIVE NOT AVAILABLE/<CR><LF>
032634  042040  044522  042526  MNRM03: .ASCIZ  @ DRIVE NOT AN RM03@<CR><LF>
032661  000040  051104  053111  MUSDR:  .ASCIZ  / DRIVE UNSAFE/<CR><LF>
032701  000040  042523  042514  MSELD:  .ASCIZ  / SELECTED/
032713  000015  050012  047522  MMODE:  .ASCIZ  <CR><LF>/PROGRAM MODE (C OR F): /
032745  000040  047506  046522  MFORMT: .ASCIZ  / FORMAT & VERIFY/
032766  044103  041505  020113  MHECK:  .ASCIZ  /CHECK ONLY/
033001  000106  051117  040515  MORMAT: .ASCIZ  /FORMAT & VERIFY/
033021  000015  005012  050117  MSIZE:  .ASCIZ  <CR><LF><LF>/OPERATE IN 32 SECTOR MODE (Y OR N) ? /
033072  050117  051105  052101  MSEC22: .ASCIZ  /OPERATION WILL BE IN 32 SECTOR (16 BIT) MODE/<CR><LF>
033151  000117  042520  040522  MSEC20: .ASCIZ  /OPERATION WILL BE IN 30 SECTOR (18 BIT) MODE/<CR><LF>
033230  047111  040526  044514  BADENT: .ASCIZ  /INVALID ENTRY/<CR><LF>
033250  047105  044504  043516  MADRER: .ASCII  /ENDING DSK ADRS MUST BE EQUAL TO OR GREATER/<CR><LF>
033325  000124  040510  020116          .ASCIZ  /THAN STARTING ADRS/<CR><LF>
```

```
033352  042523  042514  052103  MSELP:  .ASCII  /SELECT DATA PATTERN (BY ENTERING 0,1 OR 2)/<CR><LF>
033426  024040  024460  055040          .ASCII  / (0) ZERO'S /<CR><LF>
033444  024040  024461  047440          .ASCII  / (1) ONE'S/<CR><LF>
033460  024040  024462  053440          .ASCIZ  / (2) WORST CASE: /

033502  047527  051522  020124  MPATD:  .ASCIZ  /WORST CASE/
033515     015  005012  052123  MSFOU:  .ASCIZ  <CR><LF><LF>/STARTING FORMAT ON DRIVE /
033552  005015  051412  040524  MSCHK:  .ASCIZ  <CR><LF><LF>/STARTING CHECK ON DRIVE /
033606  005015  043012  051117  MFCMPT: .ASCIZ  <CR><LF><LF>/FORMAT COMPLETE, /
033633     015  005012  044103  MCCMPT: .ASCIZ  <CR><LF><LF>/CHECK COMPLETE, /
033657     124  052117  046101  NUMERR: .ASCIZ  /TOTAL ERRORS DETECTED: /
033707     120  042522  042523  ADDRIS: .ASCIZ  /PRESENT ADDRESS IS: /
033734  005015  047520  042527  PMSG:   .ASCIZ <CR><LF>/POWER ON ,PROGRAM STARTS AT 200/<LF>
033777     105  052116  051105  MESG1:  .ASCIZ  /ENTER THE SERIAL NUMBER (MAXIMUM 10 OCTAL DIGITS) /<CR><LF><LF>
034065     015  020012  047123  SN1:    .ASCIZ  <CR><LF>/ SN #    :  /
034103     015  047412  042526  MESG3:  .ASCII  <CR><LF>/OVER 126 BAD SECTORS HAVE BEEN DETECTED/<CR><LF>
034156  040520  045503  047040          .ASCIZ  /PACK NOT ACCEPTABLE !/
034204  005015  047111  047503  MESG2:  .ASCII  <CR><LF>/INCORRECT MANUFACTURE DEFINED BAD SECTOR/
034256  044440  043116  051117          .ASCII  / INFORMATION/<CR><LF>
034274  005015  047111  052111          .ASCII  <CR><LF>/INITIALIZE THE BAD SECTOR FILE ?/
034336  040450  051516  042527          .ASCIZ  /(ANSWER:  Y OR N <CR>)/
034365     015  040412  044514  MESG4:  .ASCIZ  <CR><LF>/ALIGNMENT PACK, PROGRAM HALT !/
                                .EVEN

(1)                             ;;*****************************************************************

                                .SBTTL  ERROR MESSAGES

(1)                             ;;*****************************************************************

034426  044122  030461  044440  EM1:    .ASCIZ  /RH11 INTERRUPT OCCURRED (RMAS=0)/
034467     125  042516  050130  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
034525     115  051501  041123  EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
034563     115  051501  041123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
034620  042101  051104  051505  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
034654  044122  030461  042040  EM6:    .ASCIZ  /RH11 DIDN'T RESPOND TO ADDRESSING/
034716  051104  053111  020105  EM10:   .ASCIZ  /DRIVE OFFLINE/
034734  042520  051522  051511  EM11:   .ASCIZ  /PERSISTENT DRIVE UNSAFE ERROR/
034772  047125  047503  051122  EM12:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
035035     123  043117  053524  EM13:   .ASCIZ  /SOFTWARE TIMEOUT/
035056  051104  053111  020105  EM14:   .ASCIZ  /DRIVE UNSAFE ERROR/
035101     103  047117  051124  EM15:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE@
035145     103  047117  051124  EM16:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE CHECK@
035217     122  052105  044522  EM17:   .ASCIZ  @RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE@
035275     104  052101  020101  EM20:   .ASCIZ  @DATA ERROR DURING WRITE CHECK@
035333     103  047117  051124  EM21:   .ASCIZ  @CONTROLLER/DRIVE ERROR VERIFYING HEADERS@
035404  042510  042101  051105  EM22:   .ASCIZ  @HEADER COMPARE ERROR VERIFYING HEADERS@
035453     103  046131  047111  EM23:   .ASCIZ  @CYLINDER FIELD IN HEADER IS NOT CORRECT@
035523     127  044522  042524  EM24:   .ASCIZ  @WRITE CHECK ERROR@

(1)                             ;;*****************************************************************

035545     122  040515  004523  DH1:    .ASCIZ  /RMAS   RMCS1   RMER1   RMER2   RMDS    RMDC    RMDA/
035613     104  044522  042526  DH2:    .ASCIZ  /DRIVE   RMDS   RMER1   RMER2   RMAS    RMCS1/
035665     104  044522  042526  DH3:    .ASCIZ  /DRIVE   REG ADR  DATA  RMCS1   RMER1   RMER2/
035736  051104  053111  020105  DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD    BAD   RMCS1 RMER1   RMER2/
```

```
036021   122  020110  042101  DH6:    .ASCIZ  /RH ADDRESS/
036034 051104  053111  020105  DH10:   .ASCIZ  /DRIVE   ERR PC  RMCS1   RMCS2   RMDS   RMER1   RMER2/
036121   122  042515  030503  DH10A:  .ASCIZ  /RMECI   RMEC2   RMWC    RMBA    RMDA    RMAS    RMLA/
036206 046522  041104  051011  DH10B:  .ASCIZ  /RMDB   RMMR1   RMDT    RMSN    RMOF  CYLINDER  TRACK/
036263   104  044522  042526  DH17:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
036332 051104  053111  020105  DH20:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
036401   122  041515  030523  DH20A:  .ASCIZ  /RMCS1   RMCS2   RMDS    RMER1   RMER2   RMMR2   RMEC1   RMEC2/
036476 046522  041527  020040  DH20B:  .ASCIZ  /RMWC    RMBA    RMDA    RMAS    RMLA    RMDB    RMMR1    RMDT/
036574 046522  047123  051011  DH20C:  .ASCIZ  /RMSN   RMOF  CYLINDER  TRACK/
036627   040  020040  020040  DH23:   .ASCII  /                EXPT'D/<CR><LF>
036657   104  044522  042526          .ASCIZ  /DRIVE   ERR PC  CYLNDR  ACTUAL HEADER/
036725   040  020040  020040  DH24:   .ASCII  /                                  MEMORY  DISK/<CR><LF>
037013   104  044522  042526          .ASCIZ  /DRIVE   ERR PC  CYLNDR  TRACK   SECTOR  DATA    DATA/
                              .LIST   BEX
5758
5759
5760 037100  001374  005540  005554  DT1:    .WORD   ATTN,RM.REG,RM.REG+14,RM.REG+42,RM.REG+12,RM.REG+34,RM.REG+6
5761 037106  005602  005552  005574
5762 037114  005546
5763 037116  001372  024046  024050  DT2:    .WORD   DDRIVE,RMERRS,RMERRS+2,RMERRS+4,ATTN,RMCS1
5764 037124  024052  001374  000000
5765 037132  001372  031036  031040  DT3:    .WORD   DDRIVE,RD.ADR,RD.WRD,RM.REG,RM.REG+14,RM.REG+42
5766 037140  005540  005554  005602
5767 037146  001372  031262  031260  DT4:    .WORD   DDRIVE,WRT.AD,WRT.WD,RD.WRD,RM.REG,RM.REG+14,RM.REG+42
5768 037154  031040  005540  005554
5769 037162  005602
5770 037164  001272          DT6:    .WORD   $RMADR
5771 037166  001220  001132  005540  DT10:   .WORD   DRIVE,$ERRPC,RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42
5772 037174  005550  005552  005554
5773 037202  005602
5774 037204  005604  005606  005542          .WORD   RM.REG+44,RM.REG+46,RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20
5775 037212  005544  005546  005556
5776 037220  005560
5777 037222  005562  005564  005566          .WORD   RM.REG+22,RM.REG+24,RM.REG+26,RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
5778 037230  005570  005572  001366
5779 037236  001370
5780 037240  001220  001132  001366  DT17:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
5781 037246  001370  001350
5782 037252  001220  001132  001366  DT20:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
5783 037260  001370  001350
5784 037264  005540  005550  005552          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42,RM.REG+40,RM.REG+44,RM.RE
5785 037272  005554  005602  005600
5786 037300  005604  005606
5787 037304  005542  005544  005546          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.RE
5788 037312  005556  005560  005562
5789 037320  005564  005566
5790 037324  005570  005572  001366          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
5791 037332  001370
5792 037334  001220  001132  037554  DT23:   .WORD   DRIVE,$ERRPC,BUFP,RBUF,RBUF+2
5793 037342  037544  037546
5794 037346  001220  001132  001366  DT24:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC,$GDDAT,RM.REG+22
5795 037354  001370  001350  001140
5796 037362  005562
5797 037364  005540  005550  005552          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42,RM.REG+40,RM.REG+44,RM.RE
5798 037372  005554  005602  005600
5799 037400  005604  005606
```

```
 5800  037404  005542  005544  005546          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.RE
 5801  037412  005556  005560  005562
 5802  037420  005564  005566
 5803  037424  005570  005572  001366          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
 5804  037432  001370
 5805
 5806  037434  000001                  DF1:    .WORD   1
 5807  037436     007     000                  .BYTE   7,0
 5808  037440  000001                  DF2:    .WORD   1
 5809  037442     006     000                  .BYTE   6,0
 5810  037444  000001                  DF3:    .WORD   1
 5811  037446     006     000                  .BYTE   6,0
 5812  037450  000001                  DF4:    .WORD   1
 5813  037452     007     000                  .BYTE   7,0
 5814  037454  000001                  DF6:    .WORD   1
 5815  037456     001     000                  .BYTE   1,0
 5816  037460  000003                  DF10:   .WORD   3
 5817  037462     007     000                  .BYTE   7.,0
 5818  037464  036121                          .WORD   DH10A
 5819  037466     007     000                  .BYTE   7,0
 5820  037470  036206                          .WORD   DH10B
 5821  037472     007     140                  .BYTE   7,140
 5822  037474  000001                  DF17:   .WORD   1
 5823  037476     005     034                  .BYTE   5,34
 5824  037500  000004                  DF20:   .WORD   4
 5825  037502     005     034                  .BYTE   5,34
 5826  037504  036401                          .WORD   DH20A
 5827  037506     010     000                  .BYTE   8.,0
 5828  037510  036476                          .WORD   DH20B
 5829  037512     010     000                  .BYTE   8.,0
 5830  037514  036574                          .WORD   DH20C
 5831  037516     004     014                  .BYTE   4,14
 5832  037520  000001                  DF23:   .WORD   1
 5833  037522     005     000                  .BYTE   5,0
 5834  037524  000004                  DF24:   .WORD   4
 5835  037526     007     034                  .BYTE   7,34
 5836  037530  036401                          .WORD   DH20A
 5837  037532     010     000                  .BYTE   8.,0
 5838  037534  036476                          .WORD   DH20B
 5839  037536     010     000                  .BYTE   8.,0
 5840  037540  036574                          .WORD   DH20C
 5841  037542     004     014                  .BYTE   4,14
 5842
 5843                                  ;;**********************************************************************
 5844
 5845                                  ;BUFFER STARTS HERE
 5846
 5847                                  ;;**********************************************************************
 5848
 5849  037544  000000  000000  000000  RBUF:   .WORD   0,0,0,0          ;BUFFER FOR HEADER CHECK
 5850  037552  000000
 5851
 5852  037554                          BUFP:                           ;FORMAT AND CHECK BUFFER STARTS HERE
 5853                                                                  ;AND WILL OVERLAY ALL REMAINING CODE IN PROGRAM
 5854
 5855                                  .NLIST  BEX
```

```
037554  005015  046412  044501  TITLE:  .ASCII  <CR><LF><LF>/MAINDEC-11-DZRMA/<CR><LF>
037601     122  030115  020063          .ASCII  @RM03 FORMATTER @<CR><LF><LF>
037623     120  047522  051107          .ASCIZ  /PROGRAM NEEDS 17 K MEMORY/<CR><LF><LF>
037660  005015  047524  023440  LOADRV: .ASCII  <CR><LF>/TO 'FORMAT' OR 'CHECK' DRIVE 0, REPLACE THE 'XXDP'/<CR><LF>
037746  040520  045503  047440          .ASCII  /PACK ON DRIVE 0 WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40,/<CR><LF>
040044  047101  020104  042522          .ASCIZ  /AND RESTART THE PROGRAM/<CR><LF>
                                .LIST           BEX
 5856
 5857
 5858                           .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
 5859                           ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
 5860                           ;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
 5861                           ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 5862                           ;REQUIRED.
 5863                           ;NOTE: THIS ROUTINE DESTROYS R0-R4
 5864                           ;CALL
 5865                           ;
 5866                           ;       JSR     PC,BUSADR
 5867                           ;       RETURN
 5868
 5869  040076  005737  001400  BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
 5870  040102  001434                  BEQ     3$              ;NO--BRANCH
 5871  040104  005037  001400          CLR     CHGADR          ;YES--CLEAR THE REQUEST FLAG
 5872  040110  012700  001272  1$:     MOV     #$RMADR,R0      ;FIRST ADDRESS
 5873  040114  104401  040212          TYPE    ,MRMCS1         ;"RMCS1="
 5874  040120  011046                  MOV     (R0),-(SP)      ;PRESENT RMCS1 ADDRESS
 5875  040122  104402                  TYPOC                   ;TYPE IT
 5876  040124  104401  032503          TYPE    ,LINSP          ;2 SPACES
 5877  040130  104411                  RDLIN                   ;GET THE ENTRY
 5878  040132  012601                  MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
 5879  040134  004537  040234          JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
 5880  040140  000763                  BR      1$              ;ERROR RET
 5881  040142  012700  001274  2$:     MOV     #$RMVEC,R0      ;CHECK VERC
 5882  040146  104401  040223          TYPE    ,MRHVEC         ;"RHVEC="
 5883  040152  011046                  MOV     (R0),-(SP)      ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
 5884  040154  104402                  TYPOC                   ;TYPE IT
 5885  040156  104401  032503          TYPE    ,LINSP          ;2 SPACES
 5886  040162  104411                  RDLIN                   ;READ THE ENTRY
 5887  040164  012601                  MOV     (SP)+,R1        ;ASCII TEXT ADDRESS
 5888  040166  004537  040234          JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
 5889  040172  000763                  BR      2$              ;ERROR RET
 5890  040174  012700  001272  3$:     MOV     #$RMADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
 5891  040200  012701  024214          MOV     #RMADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
 5892  040204  012021                  MOV     (R0)+,(R1)+     ;BUS ADDRESS
 5893  040206  012021                  MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
 5894  040210  000207                  RTS     PC              ;RETURN
 5895
 5896  040212  046522  051503  020061  MRMCS1: .ASCIZ  @RMCS1 = @
 5897  040220  020075     000
 5898  040223     122  053110  041505  MRHVEC: .ASCIZ  @RHVEC = @
 5899  040230  036440  000040
 5900
 5901                           .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
 5902                           ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
 5903                           ;AND FORMS AN OCTAL NUMBER IN R2
 5904                           ;CALL:
```

```
                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                ;       R0=ADDRESS TO STORE THE NUMBER
                                ;       JSR     R5,CK.NUM
                                ;       RET1    ERROR RETURN
                                ;       RET2    NORMAL RET

040234  010246          CK.NUM: MOV     R2,-(SP)        ;SAVE R2
040236  010346                  MOV     R3,-(SP)                ;SAVE R3
040240  010446                  MOV     R4,-(SP)        ;SAVE R4
040242  012703  000006          MOV     #6,R3           ;MAX OCTAL DIGITS IN THE NUMBER
040246  005002                  CLR     R2              ;FINAL OCTAL VALUE
040250  112104          1$:     MOVB    (R1)+,R4        ;GET CURRENT POINTED BYTE
040252  001424                  BEQ     3$              ;BRANCH,IF TERMINATOR DETECTED
040254  120427  000060          CMPB    R4,#'0          ;SMALLER THAN ASCII-0 ?
040260  103425                  BLO     5$              ;YES,ERROR EXIT
040262  120427  000067          CMPB    R4,#'7          ;LARGER THAN ASCII-7 ?
040266  101022                  BHI     5$              ;YES,ERROR EXIT
040270  006302                  ASL     R2              ;SHIFT LEFT
040272  103420                  BCS     5$              ;
040274  006302                  ASL     R2              ;ONE
040276  103416                  BCS     5$              ;
040300  006302                  ASL     R2              ;OCTAL DIGIT
040302  103414                  BCS     5$              ;ERROR IF CARRY BIT SET
040304  042704  177770          BIC     #177770,R4      ;CHOP OFF HIGHER BITS
040310  060402                  ADD     R4,R2           ;APPENDING CURRENT DIGIT
040312  005303                  DEC     R3              ;DECREMENT BYTE COUNT
040314  001401                  BEQ     2$              ;BRANCH,IF LAST DIGIT
040316  000754                  BR      1$              ;LOOPING BACK
040320  112104          2$:     MOVB    (R1)+,R4        ;CHECK TERMINATOR
040322  001004                  BNE     5$              ;BRANCH IF NOT FOUND
040324  005702          3$:     TST     R2              ;FINAL VALUE = 0 ?
040326  001401                  BEQ     4$              ;YES
040330  010210                  MOV     R2,(R0)         ;REPLACE THE ORIGINAL VALUE
040332  005725          4$:     TST     (R5)+           ;ADJUST FOR NORMAL RET
040334  012604          5$:     MOV     (SP)+,R4        ;RESTORE 4
040336  012603                  MOV     (SP)+,R3        ;RESTORE R3
040340  012602                  MOV     (SP)+,R2        ;RESTORE R2
040342  000205                  RTS     R5              ;EXIT

000001                  .END
```

```
ABASE = 176700          1078#   1389    1430
ACDW1 = 000000          1389    1432
ACDW2 = 000000          1389    1433
ACK   = 000123          1326#
ACPUOP= 000000          1389    1404
ACTDRV  024132          4465#   4733*   4784*   5102*   5111*   5349
ACTSTR  024133          4471#   5351*   5365*
ADDRIS  033707          3109    5757#
ADDW0 = 000000          1389
ADDW1 = 000000          1389
ADDW10= 000000          1389
ADDW11= 000000          1389
ADDW12= 000000          1389
ADDW13= 000000          1389
ADDW14= 000000          1389
ADDW15= 000000          1389
ADDW2 = 000000          1389
ADDW3 = 000000          1389
ADDW4 = 000000          1389
ADDW5 = 000000          1389
ADDW6 = 000000          1389
ADDW7 = 000000          1389
ADDW8 = 000000          1389
ADDW9 = 000000          1389
ADEVCT= 000000          1389    1395
ADEVM = 000000          1389    1431
ADRTBL  005752          1580#   2568    2639    2817
AENV  = 000000          1389    1400
AENVM = 000000          1389    1401
AFATAL= 000000          1389    1392
AMADR1= 000000          1389    1417
AMADR2= 000000          1389    1421
AMADR3= 000000          1389    1424
AMADR4= 000000          1389    1427
AMAMS1= 000000          1389    1411
AMAMS2= 000000          1389    1419
AMAMS3= 000000          1389    1422
AMAMS4= 000000          1389    1425
AMSGAD= 000000          1389    1397
AMSGLG= 000000          1389    1398
AMSGTY= 000000          1389    1391
AMTYP1= 000000          1389    1412
AMTYP2= 000000          1389    1420
AMTYP3= 000000          1389    1423
AMTYP4= 000000          1389    1426
AOE   = 001000          1228#
APASS = 000000          1389    1394
APRIOR= 000000          1389
APT     007312          1944#
APTCSU= 000040          3608    3715#
APTENV= 000001          3489    3601    3671    3713#
APTSIZ= 000200          1851    3712#
APTSPO= 000100          3603    3673    3714#
ASWREG= 000000          1389    1402
ATA   = 100000          1215#
ATABIT  024202          1927    4537#   4700    4796    4902    5272    5291    5299    5300    5320    5406
```

```
ATESTN= 000000           1389    1393
ATTN    001374           1474#   3466*   5760    5763
ATO   = 000001           1248#
AT1   = 000002           1249#
AT2   = 000004           1250#
AT3   = 000010           1251#
AT4   = 000020           1252#
AT5   = 000040           1253#
AT6   = 000100           1254#
AT7   = 000200           1255#
AUNIT = 000000           1389    1396
AUSWR = 000000           1389    1403
AVECT1= 000254           1079#   1389    1428
AVECT2= 000000           1389    1429
A16   = 000400           1152#
A17   = 001000           1153#
BADENT  033230           3151    5757#
BAD16   001414           1487#   2236    2256    2291*   2292*   2301    2302*   2303*   2304*   2305*   2306*   2307*   2308*
                         2309*   2311*   2313*   2315*   2316*   2317*   2318*   2319*   2320*   2323*   2329    2331    2333
                         2334    2335    2336    2337    2338    2341*   2428    3245    3258
BAD18   002420           1490#   2235    2262    2333*   2338*   2342*   2431    3261
BAI   = 000010           1170#
BEGCYL  001322           1450#   1561    1955*   2045*   2101*   2109    2349    2358    2362    2716    2720    2762    2765*
                         2768*   2937    2966    2968*   2975*   2997    3231    3237*   3243*
BEGIN   006412           1115    1811#
BEGIN1  006422           1114    1813#   1858
BEGIN2  006436           1812    1817#   4156
BEGTRK  001326           1452#   1562    1954*   2044*   2100*   2113    2352    2365    2763    2764*   2767*   2936    2939
                         2967    2969*   2974*   2999    3232    3238*   3242*
BIT0  = 000001           1063#
BIT00 = 000001           1053#   1063    1930    2097
BIT01 = 000002           1052#   1062    1933    2097    2922    4776    5024
BIT02 = 000004           1051#   1061
BIT03 = 000010           1050#   1060    5231    5524
BIT04 = 000020           1049#   1059    2903    5266
BIT05 = 000040           1048#   1058    4627    5044    5246    5401
BIT06 = 000100           1047#   1057    4709    4767    4841    5142    5585
BIT07 = 000200           1046#   1056    4709    4976    5123    5231    5246    5266    5295    5557
BIT08 = 000400           1045#   1055    2900    4709
BIT09 = 001000           1044#   1054    2900    3500    5399
BIT1  = 000002           1062#   2677    2891
BIT10 = 002000           1043#   2897    2922    3477    5026
BIT11 = 004000           1042#   2897    4674    4803    4995    5335
BIT12 = 010000           1041#   2209    2894    2922    4669    4691    4709    4778    4811    4991    5010    5022    5035
                         5256    5258    5462    5519    5586
BIT13 = 020000           1040#   2891    2922    3484    4761    5204    5465
BIT14 = 040000           1039#   2207    2682    2686    2688    2891    2922    4773    4808    5191    5331    5427    5474
                         5478    5583
BIT15 = 100000           1038#   2207    2302    2320    2512    2525    2528    2543    2627    2710    4761    4773    4776
                         4778    4808    4811    4995    5024    5026    5142    5231    5246    5256    5266    5331    5399
                         5427    5434
BIT2  = 000004           1061#   2891    5434
BIT3  = 000010           1060#
BIT4  = 000020           1059#
BIT5  = 000040           1058#
BIT6  = 000100           1057#
```

```
BIT7   = 000200   1056#
BIT8   = 000400   1055#
BIT9   = 001000   1054#
BPTVEC= 000014   1070#
BUFP    037554   1580  1581  1582  1583  1584  1585  1586  1587  1588  1589  1590  1591  1592
                 1593  1594  1595  1596  1597  1598  1599  1600  1601  1602  1603  1604  1605
                 1606  1607  1608  1609  1610  1611  2169  2207  2209  2211* 2212  2214  2216
                 2218  2220  2222  2225  2255  2261  2267  2270  2392  2498  2743  2798  2934
                 2996  3015  3034  3224  3246  3259  3262  3265  3268  3271  3274  5792  5852#
BUSADR  040076   1877  5869#
C       032473   3110  5757#
CHGADR  001400   1477# 1811* 1813* 5869  5871*
CI1     025466   4831  4859#
CI3     025574   4833  4882#
CI4     025674   4822  4904#
CI5     026252   4881  4903  4981# 4990
CI6     026274   4920  4930  4932  4934  4986#
CI7     026310   4763  4840  4871  4875  4879  4887  4897  4901  4912  4919  4925  4929  4943
                 4949  4953  4963  4974  4989  4991# 5097  5127  5283  5569
CI7B    026336   4994  4997# 5285
CI8     026410   4992  5000  5009# 5157  5419
CKADRS  011744   2349# 2712  2867  2870
CKSWR  = 104407   3464  3499  4353#
CKTRK   013334   2596#
CK.CHR  017656   2060  2130  3370# 3407  3421
CK.DEC  017630   3348# 3376
CK.DIG  017730   3142  3401#
CK.NUM  040234   5879  5888  5913#
CK.OCT  017602   3330# 3378
CLOCK   016312   3048  3058  3085#
CLR    = 000040   1172#
CLRQUE  032046   4609  5043  5653#
CNTLC   001376   1476# 1858* 2054* 3108* 3905  3988
CR     = 000015    978# 1436# 3647  3657  5757  5855
CRLF   = 000200    979# 3618  3657
CYLCK   001344   1459# 2716* 2747  2750* 2751  2755* 2756  2758*
DCK    = 100000   1234# 2620
DDISP  = 177570    985# 1367  1839
DDRIVE  001372   1473# 3465* 5763  5765  5767
DEVMP   001404   1480# 1927* 1949* 1950* 1951  1981  1989* 2832  2858
DF1     037434   1670  5806#
DF10    037460   1719  1726  1733  1740  1747  1754  1761  1782  1789  5816#
DF17    037474   1768  5822#
DF2     037440   1677  5808#
DF20    037500   1775  5824#
DF23    037520   1796  5832#
DF24    037524   1803  5834#
DF3     037444   1684  5810#
DF4     037450   1691  5812#
DF6     037454   1705  5814#
DH1     035545   1668  5757#
DH10    036034   1717  1724  1731  1738  1745  1752  1759  1780  1787  5757#
DH10A   036121   5757# 5818
DH10B   036206   5757# 5820
DH17    036263   1766  5757#
DH2     035613   1675  5757#
```

```
DH20     036332  1773  5757#
DH20A    036401  5757#  5826  5836
DH20B    036476  5757#  5828  5838
DH20C    036574  5757#  5830  5840
DH23     036627  1794  5757#
DH24     036725  1801  5757#
DH3      035665  1682  5757#
DH4      035736  1689  5757#
DH6      036021  1703  5757#
DISPLA   001156  1367#  1839*  1847*  3476*
DISPRE   000174  1100#  1847
DLT    = 100000  1182#
DMD    = 000001  1238#
DONE     015102  2855  2856#
DPINT    024106  4425#  4638  4641  4717*  4743  5175  5200  5321  5323*  5390  5408  5413  5421*
DPR    = 000400  1208#
DPRQS    024116  4438#  4749  4797*  4835*  5178  5392  5410  5415  5430*
DRIVE  = 001220  1445#  1988*  2072*  2093  2095  2156  2167  2931  3301  5771  5780  5782  5792
                 5794
DRQ    = 004000  1271#
DRVACT   024056  4384#  4757  4937*  4984*  5003*  5014  5029*  5116*  5196  5229  5260*  5267*  5280
                 5290*  5402#
DRVCLR=  000111  1321#
DRVINT   024444  4629  4663#  4745  5311  5324
DRVQUE   032144  4752  4765  5693#
DRVSTA   024066  1912  1990  2082  4398#  4619*  4620*  4621*  4622*  4632*  4664*  4673*  4706*  4711*
                 4739  4747  4771  4805  4809  4938*  5047*  5181  5189  5202  5254*  5261*  5273
                 5326  5422*
DRVTYP   024076  1915  1930  1933  2087  2097  4412#  4665*  4680*  4685*  4774  5048*
DRY    = 000200  1207#
DSWR   = 177570   984#  1366  1838
DS.CYL   001366  1471#  3467*  5777  5780  5782  5790  5794  5803
DS.TRK   001370  1472#  3468*  5777  5780  5782  5790  5794  5803
DTE    = 010000  1231#  2620
DTSY   = 000200  1244#
DTUW     024200  4531#  4615  4824  4880*  5004  5007*  5017  5030  5032*  5041*  5105  5117*  5395
                 5472
DT00   = 000001  1262#
DT01   = 000002  1263#
DT02   = 000004  1264#
DT03   = 000010  1265#
DT04   = 000020  1266#
DT05   = 000040  1267#
DT06   = 000100  1268#
DT07   = 000200  1269#
DT08   = 000400  1270#
DT1      037100  1669  5760#
DT10     037166  1718  1725  1732  1739  1746  1753  1760  1781  1788  5771#
DT17     037240  1767  5780#
DT2      037116  1676  5763#
DT20     037252  1774  5782#
DT23     037334  1795  5792#
DT24     037346  1802  5794#
DT3      037132  1683  5765#
DT4      037146  1690  5767#
DT6      037164  1704  5770#
```

```
DULP    024446          4664#   4675
DVA   = 004000          1202#
ECH   = 000100          1225#
ECI   = 004000          1293#
EDIT    001410          1482#   1946*   2372*   2375*   2782
EMPTYQ  032124          5001    5046    5143    5268    5333    5435    5678#
EMTVEC= 000030          1073#   1826*   1827*   1857*
EM1     034426          1667    5757#
EM10    034716          1716    5757#
EM11    034734          1723    5757#
EM12    034772          1730    5757#
EM13    035036          1737    5757#
EM14    035056          1744    5757#
EM15    035101          1751    5757#
EM16    035145          1758    5757#
EM17    035217          1765    5757#
EM2     034467          1674    5757#
EM20    035275          1772    5757#
EM21    035333          1779    5757#
EM22    035404          1786    5757#
EM23    035453          1793    5757#
EM24    035523          1800    5757#
EM3     034525          1681    5757#
EM4     034563          1688    5757#
EM5     034620          5757#
EM6     034654          1702    5757#
ENDCYL  001320          1449#   1563    1956*   2096*   2099*   2109    2349    2359    2361    2747    2756
ENDTRK  001324          1451#   1564    1957*   2047*   2113    2351    2366
ENTADR  032506          2106    5757#
ERINDX  015230          2192    2579    2611    2731    2888#
ERR   = 040000          1214#   2631
ERRVEC= 000004          1066#   1836    1837*   1848*   3044*   3045*   3055*   3064*
ES.SAV  032310          5160    5211    5215    5470    5530    5748#
FER   = 000020          1223#   2620
FMTDPB  005610          1530#   2049*   2093    2095*   2167*   2168*   2169*   2170*   2171*   2172*   2173*   2175*   2176*
                        2178    2180    2182*   2186    2188    2205    2404*   2407*   2408*   2410    2412    2438    2440
                        2496*   2497*   2498*   2508*   2516    2527*   2529    2531    2548    2553*   2559    2564*   2566
                        2568*   2569*   2573    2575    2596*   2600    2602    2637*   2639*   2647    2652*   2661    2717*
                        2718*   2719*   2720*   2721*   2725    2727    2737    2753    2759*   2760*   2761*   2771*   2787*
                        2788*   2790    2792    2794*   2795*   2796*   2797*   2798*   2799*   2801    2803    2809    2814*
                        2817*   2818*   2819*   2890    2893    2896    2899    2902    2921    2931*   2932*   2933*   2934*
                        2935*   2936*   2937*   2938*   2951*   2952    2954    2962*   2963*   2964    2971*   2972*   2973*
                        2984*   2985*   2986*   3111    3117    3167    3169    3171    3177    3178    3297    3298    3306
                        3307    3467    3468
FMTX    005630          1547#   3301*   3302*   3303*   3304*   3305*   3306*   3307*   3313    3315
FMT16 = 010000          1294#
F1    = 000002          1197#
F2    = 000004          1198#
F3    = 000010          1199#
F4    = 000020          1200#
F5    = 000040          1201#
GETREG= 000141          1328#
GETREQ  032220          4798    5019    5134    5185    5221    5282    5330    5398    5424    5431    5714#
GNS   = ****** U        1099    4345    4346    4347    4348    4349    4351    4353    4354    4355    4356    4357
GO    = 000001          1196#
GTSWR = 104406          1871    4351#
```

```
HCE     = 000200          1226#   2620    2737
HCI     = 002000          1292#
HCRC    = 000400          1227#   2620
HDREAD    014076          2696    2703#
HEADX     001412          1483#   2627*   2682*   3300
HEDERR    001360          1465#   2713*
HT      = 000011           976#   3616    3657
IAE     = 002000          1229#
IE      = 000100          1150#
ILF     = 000001          1219#
ILR     = 000002          1220#
IOTVEC= 000020            1071#
IR      = 000100          1173#
ISR       027020          4624    5102#
LA        026644          4828    5063#
LACNT     024144          4490#   5071*   5072    5093*
LF      = 000012           977#   1437#   3651    3657    5757    5855
LINSP     032503          1928    3114    3553    3562    3573    5757#   5876    5885
LIN4SP    032501          1911    5757#
LOADRV    037660          5855#
LOP.CK    015330          2198    2586    2666    2742    2746    2917#
LST     = 002000          1210#
MADRER    033250          2115    5757#
MAXSEC    001364          1469#   1968*   1973*   2037*   2043*   2521    2531    2539    2556    2607    2640    2661    2994
                          3014    3033    3233    3235*   3241*
MCCMPT    033633          2827    5757#
MCLK    = 000002          1239#
MCPE    = 020000          1155#
MCPEMX    024212          4549#   5452    5500
MDRNP     032560          2089    5757#
MDRVD     032477          2068    5757#
MER11     032605          2091    5757#
MESG1     033777          2288    5757#
MESG2     034204          3072    5757#
MESG3     034103          3182    5757#
MESG4     034365          2876    5757#
MFCMPT    033606          2825    5757#
MFORMT    032745          2015    5757#
MHECK     032766          2013    5757#
MINX    = 000004          1240#
MMODE     032713          2002    5757#
MNDLTA    024226          4565#   5091
MNRM03    032634          5757#
MODE      001316          1448#   1959*   1962*   1974    2001*   2018*   2122    2150    2379    2703    2823    2865
MOFFLN    032536          5757#
MOH     = 020000          1272#
MOL     = 010000          1212#
MORMAT    033001          2017    5757#
MPATD     033502          2137    5757#
MPE     = 000400          1175#
MRD     = 000020          1242#
MRHVEC    040223          5882    5898#
MRMCS1    040212          5873    5896#
MSCHK     033552          2154    5757#
MSEC20    033151          2039    5757#
MSEC22    033072          2033    5757#
```

```
MSELD   032701          5757#
MSELP   033352          2127    5757#
MSFOU   033515          2152    5757#
MSIZE   033021          2022    5757#
MSTCK = 000010          1241#
MUNIT   032455          2057    5757#
MUSOR   032661          2085    5757#
MWC     001356          1464#   1967*   1971*   2035*   2041*   2497    2933
MWR   = 000040          1243#
MXDLTA  024224          4562#   5089
MXF   = 001000          1176#
MXLACT  024222          4559#   5072
MXWNDW  024230          4568#
M0      010106          2050    2054#   2071    2086    2090    2092    2863
M1      007630          1945    2001#   2012
M1A     007720          2014    2022#   2032
M1B     010046          2038    2044#   3095
M2      010356          2094    2102    2106#   2116
M4      010434          2112    2114    2118    2122#   2143
M4A     010460          2123    2127#
M5      010542          1994    2126    2148#
M5FX    007360          1952    1954#   2860
NBA   = 100000          1274#
NED   = 010000          1179#
NEM   = 004000          1178#
NEWSVC  016344          2148    2345    3099#   3121
NEXT    005536          1501#   2493*   2505    2508    2524*   2542*   2544    2555*   2641    2643    2674
NOTPRS  032372          1920    5757#
NOTRM   032355          1918    5757#
NOTSAF  032407          1924    5757#
NUMERR  033657          2828    5757#
OENTER  016324          2054    3091#   3108
OFDIR = 000001          1291#
OFFSET= 000115          1323#
OPI   = 020000          1232#   2620
OPT     025230          4759    4794#   5137    5304    5341
OR    = 000200          1174#
PACK    001406          1481#   2174*   2227*   2253    2282    2426    2451*   2534*
PAR   = 000010          1222#
PARENT  016502          2107    3129#
PAR1    005702          1561    1569#
PAR2    005715          1562    1571#
PAR3    005730          1563    1573#
PAR4    005741          1564    1575#
PAT   = 000020          1171#
PATA    001340          1457#   1976*   2124*   2381*   2385*   2389*   2393    2705    2708*   2710*   2711    2868*   3020
PATB    001342          1458#   1977*   2125*   2382*   2386*   2390*   2394    2711*   2869*   3021
PATSEL  001336          1456#   1978*   2048*   2144*   2383    2387
PGE   = 002000          1177#
PGM   = 001000          1209#
PIP   = 020000          1213#
PIRQ  = 177772           983#
PIRQVE= 000240          1077#
PMSG    033734          4154    5757#
POPQUE  032242          4807    4859    4975    5294    5729#
PRO   = 000000          1000#
```

```
PR1    = 000040          1001#
PR2    = 000100          1002#
PR3    = 000140          1003#
PR4    = 000200          1004#
PR5    = 000240          1005#  1856   1857
PR6    = 000300          1006#  3050   3060
PR7    = 000340          1007#  1901
PS     = 177776           980#   981   1901*  2081*  4607   4608*  4637*  4645*  4731   4732*  4785*  4795   4844*
                         5078*
PSEL   = 002000          1154#
PSW    = 177776           981#  2149*  2346*  3107*
PWRVEC= 000024          1072#  1830*  1831*  4123*  4124*  4133*  4139*  4151*  4152*
QCNT    031554          5595#  5654   5678*  5693   5695*  5715   5729*
QDRV0   031646          5606   5617   5626   5638#
QDRV1   031666          5607   5618   5627   5639#
QDRV2   031706          5608   5619   5628   5640#
QDRV3   031726          5609   5620   5629   5641#
QDRV4   031746          5610   5621   5630   5642#
QDRV5   031766          5611   5622   5631   5643#
QDRV6   032006          5612   5623   5632   5644#
QDRV7   032026          5613   5624   5633   5645#
QINPT   031564          5606#  5680   5697*  5698*  5699   5701*
QOUTPT  031604          5617#  5680*  5718   5731   5732*  5733*  5734   5736*
QSTART  031624          5626#  5660   5665   5701   5736
QSTOP   031626          5627#  5699   5734
QTERM  = 032046          5634   5646#
RBUF    037544          1552   2718   2743   3297*  3298*  3299*  3300*  3304   3308*  3311*  5792   5849#
RDCHR  = 104410          4064   4354#
RDDAT  = 000171          1335#
RDHD   = 000173          1336#  2173   2182   2721
RDLIN  = 104411          2003   2023   2058   2128   2293   3073   3140   4355#  5877   5886
RDY    = 000200          1151#
RD.ADR  031036          5454*  5455*  5457#  5765
RD.RM   031012          4676   4695   4701   4923   4947   4961   5065   5079   5125   5155   5223   5233   5248
                        5385   5452#  5507   5521   5527   5561
RD.RM1  031034          5456#  5486
RD.RM2  031160          5452*  5485#
RD.RM3  031164          5463   5477   5487#
RD.RM4  031170          5468   5489#
RD.WRD  031040          5458#  5459   5765   5767
READIN= 000121          1325#
RECAL  = 000107          1320#
RECORD  016616          2626   3164#
REJECT  015212          2224   2875#
RELSE  = 000113          1322#
RESET   017436          2628   2690   3297#
RESREG= 104413          3575   4244   4357#  4646   4779   4782   4845   5050   5110   5364   5570   5669
RESTRT  016246          2450   2533   3069#
RESVEC= 000010          1067#
RETBAD  010610          2166#
RETRY   001346          1460#  2166*  2199   2201*  2204*  2495*  2587   2589*  2592*  2667   2669*  2672*  2784*
                        2806*  2807   2820*
RMADR   024214          1878*  1888*  4554#  4626   4738   4862   4882   4904   5063   5104   5383   5454   5460
                        5464   5480   5513   5517   5550   5580   5891
RMAS   = 000016          4579#  4700*  5153   5272*  5299*  5320*  5405
RMBA   = 000004          3469   3471   4574#
```

```
RMCS1 = 000000   4572# 4668* 4674  4689  4802* 4803  4878  4900  4942  4962  4973  4988  4999
                 5126  5156  5208  5240  5277  5335  5508  5763
RMCS2 = 000010   2617  4576# 4627* 4667* 4669  4751* 4801* 4863* 4883* 4905* 4991  5010  5022
                 5035  5044* 5064* 5124* 5222* 5306* 5319* 5384* 5401* 5461  5518  5551* 5557
                 5581* 5586
RMDA  = 000006   2605  2811  4575# 4870  4896  4911
RMDB  = 000022   4581# 5555
RMDC  = 000034   4586# 4874  4886  4918
RMDS  = 000012   2631  4577# 4696  5066  5224  5249  5307  5386
RMDT  = 000026   4583# 4677
RMEC1 = 000044   4590#
RMEC2 = 000046   4591# 5565
RMERRS 024046    4376# 4610  5191  5194  5204  5307* 5308* 5309* 5310* 5763
RMER1 = 000014   2619  2924  4578# 4702  5234  5308  5522
RMER2 = 000042   4589# 5309
RMHR  = 000036   4587#
RMINIT 024232    1895  2078  4606#
RMLA  = 000020   4580# 5080  5082
RMMR1 = 000024   4582#
RMMR2 = 000040   4588# 5310
RMOF  = 000032   4585# 4693  4924  4928  4948  4952
RMR   = 000004   1221#
RMSN  = 000030   4584#
RMVEC  024216    1879* 1889* 4555# 4623  4625  4732
RMWC  = 000002   4573#
RM.REG 005540    1506# 1539  2605  2617  2619  2631  2811  2924  3469  3471  5760  5765  5767
                 5771  5774  5777  5784  5787  5790  5794  5797  5800  5803
RM03   024736    2177  2185  2409  2572  2599  2724  2789  2800  3312  4731#
RM03A  032431    1935  5757#
RM03B  032417    1929  5757#
RMOP  = 000101   1318#
RPTMR  030404    3086  5349#
RPOS   032424    1932  5757#
RTC   = 000117   1324#
SAVEFG 024154    1896* 2079* 4505# 4977  5130  5296
SAVREG= 104412   3515  4218  4356# 4606  4734  4794  5009  5103  5352  5547  5653
SAVSEC 001350    1461# 2605* 2607* 2608* 2609* 2636  2637  2689* 2811* 2813* 2814  2815  3179
                 3299  3305  5780  5782  5794
SAVWC  001352    1462# 2651* 2984
SC     027234    5109  5138  5141  5147  5153#
SCAWC  015662    2663  2984#
SC01  = 000100   1278#
SC02  = 000200   1279#
SC04  = 000400   1280#
SC10  = 001000   1281#
SC11   030066    5193  5230  5287#
SC12   030176    5187  5198  5306#
SC13   030246    5177  5180  5316#
SC20  = 002000   1282#
SC3    027304    5168# 5172
SC4    027310    5170# 5201  5213  5217  5265  5279  5286  5305  5342
SC5    027322    5169  5175#
SC6    027522    5197  5218#
SC6A   027632    5195  5242#
SC7    027760    5232  5245  5247  5257  5267#
SC8    030036    5209  5225  5235  5241  5250  5278  5280# 5315
```

```
SEARCH= 000131  1327#
SECCU   005434  1499#  2417   2434   2458   2459   2476   2478*  2487   2489   2509   2683
SEC30   001362  1466#  1963*  1972*  2036*  2042*  2268   2405   2429   3001   3234   3236*  3240*  3266
                3272   3309
SEEK  = 000105  1319#
SEEKFG  024156  2080*  4513#  4611   4826
SELDRV= 000145  1330#
SERNL   011364  2282#
SETFMT= 000143  1329#  2175   2408   2788
SETHDR  015710  2403   2715   2766   2970   2994#  3239
SETPAT  012116  2354   2374   2379#
SETTBL  015376  2402   2714   2931#
SETVEC  007050  1876   1881   1894#
SET.IE  031502  4640   4671   4769   4843   5049   5162   5579#
SKI   = 040000  1304#
SLASH   032467  3139   5757#
SN1     034065  2289   5757#
SOFSW   001314  1447#  2494*  2623   2653*  2655   2657   2659*  2664*  2673*  2987*
SORT1   012464  2458#
SORT2   016732  2785   3189#
SRCHWT  024130  4459#  4796*  4902*  5291   5300*
STACK = 001100   971#  1824   2184   2232   2571   2598   2723   2864   3091
STCLK1  016162  3044   3054#
STCLK2  016232  3055   3063#
STCLK3  016236  3053   3062   3064#
STKLMT= 177774   982#
ST0     030474  5359   5379#
ST01    030530  5390#
ST02    030620  5389   5391   5393   5405#
ST03    030650  5407   5413#
ST05    030674  5387   5419#
ST06    030702  5409   5421#
ST07    030740  5411   5429#
ST08    030770  5428   5435#
ST09    031000  5397   5404   5412   5416   5418   5420   5426   5433   5437#
ST.CLK  016104  1894   2077   3044#
SVRH70  031364  4979   4996   5027   5132   5144   5186   5237   5255   5298   5332   5400   5436   5547#
SWR     001154  1366#  1822   1838*  1840   1846*  1853*  1869   1899   2677   2861   2917   3477   3484
                3496   3500   3908   3947   4002*  4131   4144*
SWREG   000176  1101#  1846   1869   3908   3947   3970
SW0   = 000001  1035#  2861
SW00  = 000001  1025#  1035
SW01  = 000002  1024#  1034
SW02  = 000004  1023#  1033   1899
SW03  = 000010  1022#  1032
SW04  = 000020  1021#  1031
SW05  = 000040  1020#  1030
SW06  = 000100  1019#  1029
SW07  = 000200  1018#  1028
SW08  = 000400  1017#  1027
SW09  = 001000  1016#  1026   2917
SW1   = 000002  1034#
SW10  = 002000  1015#
SW11  = 004000  1014#
SW12  = 010000  1013#
SW13  = 020000  1012#
```

F10

CROSS REFERENCE TABLE -- USER SYMBOLS

```
SW14  = 040000          1011#
SW15  = 100000          1010#
SW2   = 000004          1033#
SW3   = 000010          1032#
SW4   = 000020          1031#
SW5   = 000040          1030#
SW6   = 000100          1029#
SW7   = 000200          1028#
SW8   = 000400          1027#
SW9   = 001000          1026#
SYSTAT  032436          1904    5757#
T       032475          3115    5757#
TABINT  011172          2232#   2452    2535
TABLD   011254          2253#
TABLE   005650          1561#   3130    3156
TAB.XY= 001114          1140#   1345
TAP   = 040000          1273#
TBITVE= 000014          1068#
TD      027064          5107    5116#
TEXT    017070          2786    3224#
TIMER   024160          4518#   4715*   4838*   4982*   5028*   5119*   5219*   5263*   5302*   5317*   5338*   5355    5357*
                        5417*   5423*   5429*
TITLE   037554          1861    5855#
TKVEC = 000060          1075#   2148*   2345*   3121*   3885*   3886*
TPVEC = 000064          1076#
TRAPVE= 000034          1074#   1828*   1829*   1856*
TRE   = 040000          1156#
TRKCNT  001334          1455#   2939*   2948    2950*   2961*
TRKTST  015466          2695    2946#
TRK1  = 004000          1283#
TRK10 = 040000          1286#
TRK2  = 010000          1284#
TRK20 = 100000          1287#
TRK4  = 020000          1285#
TRNSWT  024126          3092    4451#   4860*   5006*   5016    5033*   5042*   5121    5122*   5394
TRTVEC= 000014          1069#
TST1    006426          1815#
TTRKS   001330          1453#   2351*   2352*   2353*   2364*   2365*   2366*   2367*   2370*   2373    2940
TTRKSC  001332          1454#   2940*   2946*   2956*
TYPADR  016402          3107#
TYPDS = 104405          2830    3138    3550    4349#
TYPE  = 104401          1861    1903    1904    1911    1918    1920    1922    1924    1926    1928    1936    1938    1942
                        2002    2011    2013    2015    2017    2022    2031    2033    2039    2057    2068    2070    2085
                        2089    2091    2106    2115    2127    2137    2142    2152    2154    2161    2287    2288    2289
                        2825    2827    2828    2831    2875    2876    3072    3109    3110    3114    3115    3120    3133
                        3139    3151    3181    3182    3479    3487    3525    3526    3530    3531    3540    3553    3557
                        3562    3564    3570    3573    3621    3777    3853    3902    3914    3968    3969    3972    3983
                        3993    4004    4023    4071    4077    4082    4086    4091    4092    4094    4097    4101    4153
                        4180    4345#   5873    5876    5882    5885
TYPERR  020322          3486    3515#
TYPOC = 104402          3546    3971    4346#   5875    5884
TYPON = 104404          4348#
TYPOS = 104403          1908    2158    4347#
ULDFLG  024134          4478#   4666*   4741    4756*   4939*   5002*   5034*   5183    5269    5271*   5287    5289*   5403*
UNS   = 040000          1233#
UNTOFF  032334          1922    5757#
```

```
UNTON   032345          1926    5757#
UPDACY  015776          2770    2977    3014#
UPDATK  016054          2959    3033#
UPE     =  020000       1180#
USSAV   004430          1496#   2234    2271    2335*   2337*   2340*   3270
USTAB   003424          1493#   2237    2334*   2336*   2343*   3164    3174    3189    3191    3215    3264
US1     =  000001       1167#
US2     =  000002       1168#
US4     =  000004       1169#
VV      =  000100       1206#
WC      001354          1463#   1966*   1970*   2034*   2040*   2391
WCE     =  040000       1181#   2617
WCF     =  000040       1224#
WCKD    =  000151       1331#
WCKHD   =  000153       1332#   2596
WCTBL   006052          1615#   2547    2550    2558    2561    2646    2649    2819
WLE     =  004000       1230#   2924
WRAP    001402          1478#   1958*   2046*   2108*   2117*   2355
WRL     =  004000       1211#
WRTDAT=  000161         1333#
WRTHD   =  000163       1334#   2569    2799    3303
WRTRK   012234          2402#
WRTRKA  012666          2505#   2676
WRTRKB  012772          2517    2527#   2530
WRTRKC  013066          2526    2544#
WRTRKD  013132          2511    2522    2537    2540    2555#
WRTRKE  013200          2554    2566#
WRTRKX  012310          2417#   2679    2697
WRTRKY  012332          2426#
WRTRKZ  014064          2695#
WRTRK1  012636          2494#
WRTRK2  013214          2569#   2654
WRT.AD  031262          5503*   5513*   5516#   5767
WRT.RM  031172          4688    4692    4869    4873    4877    4885    4895    4899    4910    4917    4927    4941    4951
                        4972    4987    4998    5207    5239    5276    5500#
WRT.R1  031256          5514#   5535
WRT.R2  031350          5500*   5534#
WRT.R3  031354          5509    5520    5523    5529    5536#
WRT.R4  031360          5525    5538#
WRT.RS  031362          5537    5539#
WRT.WD  031260          5501*   5505    5512*   5515#   5767
SAPTHD  001100          1127    1133#
SASTAT=  ******  U      3693    3708
SATYC   021060          3664    3666#
SATY1   021034          3662#
SATY3   021042          3606    3663#
SATY4   021052          3492    3665#
SAUTOB  001150          1363#   1873*   1944    2284    2856    3069    3961    4118
SBASE   001262          1430#   1885    1887
SBDADR  001136          1358#
SBDDAT  001142          1360#
SBELL   001176          1377#   3479    3508    3914    4111
SCDW1   001266          1432#   1960
SCDW2   001270          1433#   1964
SCHARC  021030          3623*   3633*   3640    3649*   3654#
SCKSWR  022260          3947#   4353
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
SCMTAG  001114          1346#   1819    1820    1832
SCM3  =  000000         1376#
SCNTLC  023163          3902    3983    4111#
SCNTLG  023175          3968    4113#
SCNTLU  023170          3993    4086    4112#
SCPUOP  001234          1404#
SCRLF   001203          1379#   1903    1938    1942    2161    2287    2831    2875    3120    3181    3487    3508    3525
                        3530    3540    3557    3570    3622    3657    4004    4091    4111
SDBLK   021744          3819    3853    3861#
SDB2D   023472          4199    4218#
SDECVL  023652          4220    4266#
SDEVCT  001216          1395#   1992*   2822*   2873*
SDEVM   001264          1431#   1947    1949
SDOAGN  015076          2842    2847    2853#
SDTBL   021734          3822    3857#
SENDAD  015066          1109    2849#
SENDCT  015052          2844#
SENV    001226          1400#   1867    1880    3489    3601    3671    3695
SENVM   001227          1401#   1851    3603    3608    3673
SEOP    014506          1953    1995    2706    2749    2781#   2926
SEOPCT  015044          2841#   2845
SERFLG  001117          1349#   3474*   3508
SERMAX  001131          1355#   1833*
SERROR  020066          1826    3463#
SERRPC  001132          1356#   3481*   3482*   3483    3508    5771    5780    5782    5792    5794
SERRTB  006152          1664#   3522
SERTTL  001126          1353#   2055*   2829    3480*   3508
SESCAP  001174          1376#   1832*   2191*   2203*   2578*   2591*   2610*   2625*   2630*   2671*   2730*   2740*   2920*
                        3503    3505    3508    5160    5211    5215    5470    5530    5740    5749    5750*   5752*
SETABL  001226          1399#
SETEND  001272          1139    1434#
SFATAL  001210          1392#   3699*
SFFLG   021300          3662*   3665*   3693    3702*   3710#
SFILLC  001172          1374#   3626    3657
SFILLS  001171          1373#   3657
SGDADR  001134          1357#
SGDDAT  001140          1359#   3473*   5794
SGET42  015056          2846#
SGTSWR  022350          3969#   4351
SHD  =  000000           968
SHIBTS  001100          1134#
SICNT   001120          1350#
SILLUP  023370          4123    4139    4158#
SINTAG  001151          1364#   3966*   3985    4005    4118
SITEMB  001130          1354#   3483*   3491    3508    3517
SLF     001204          1380#   3508    3657    4101    4111
SLFLG   021277          3703*   3709#
SLKCSB  001300          1441#   3051*
SLKCSR  001276          1440#   3046    3052*
SLKS    001306          1443#   3056    3061*
SLLVEC  001310          1444#   3057
SLPADR  001122          1351#
SLPERR  001124          1352#   2183*   2570*   2597*   2722*   2919    3502
SLPVEC  001302          1442#   3047
SMADR1  001240          1417#
SMADR2  001244          1421#
```

```
SMADR3  001250            1424#
SMADR4  001254            1427#
SMAIL   001206            1135    1139    1390#   1816    1850    1867    3489    3601
SMAMS1  001236            1411#
SMAMS2  001242            1419#
SMAMS3  001246            1422#
SMAMS4  001252            1425#
SMBADR  001102            1135#
SMFLG   021276            3663*   3669    3704*   3708#
SMNEW   023213            3972    4116#
SMSGAD  001222            1397#   3679*   3682
SMSGLG  001224            1398#   3684*
SMSGTY  001206            1391#   3677    3685*   3697    3701*
SMSWR   023202            3969    4114#
SMTYP1  001237            1412#
SMTYP2  001243            1420#
SMTYP3  001247            1423#
SMTYP4  001253            1426#
SNULL   001170            1372#   3628    3657
SNWTST= 000000            1814#
SOCNT   021524            3749*   3778*   3791#
SOMODE  021526            3744*   3748*   3753    3756*   3767*   3793#
SPASS   001214            1394#   1850*   2834*   2838*   2839*   2856    2871*
SPASTM  001106            1137#
SPWRAD  023364            4156#
SPWRON  023224            1830    4123#   4151
SPWRMG  023360            4154#
SPWRUP  023276            4133    4139#
SQUES   001202            1378#   2011    2031    2070    2142    3508    3657    4023    4094    4111
SRDCHR  022622            4036#   4354
SRDDEC= ****** U          4356
SRDLIN  022712            4059#   4355
SRDOCT= ****** U          4356
SRDSZ = 000015            4052#
SRESRE  023724            4301#   4357
SRMADR  001272            1438#   1878    1887*   1888    5770    5872    5890
SRMVEC  001274            1439#   1879    1884*   1889    5881
SRTNAD  015100            2855#
SR2A  = ****** U          4358
SSAVRE  023666            4285#   4356
SSAVR6  023374            4132*   4140    4141*   4142*   4160#
SSB2D   023436            3112    3118    4197#
SSETUP= 000116            1811#   1825    1826    1828    1830    1832    1834    1864    2838    3464    3499    3507    3906
                          3911    3912    3942    4118
SSTUP = 177777            1811#
SSUPRS  023376            3113    3119    4172#
SSVPC = 000200            1107#   1112
SSWR  = 123000             958#    968    1085    1086    1087    1088    1089    1376    1377    1816    1832    2778    2838
                          2848    2854    2856    3455    3456    3457    3458    3459    3477    3484    3496    3500    3508
                          4157
SSWREG  001230            1402#   1853
STESTN  001212            1393#   1816*   2872*   2995*   3027*   3230    3244*
STKB    001162            1369#   3099    3103    3866    3887    3898    3923    3951    3978
STKCNT  021754            3867#   3882*   3912    3929*   4043    4045*
STKINT  021772            1863    2056    2286    3071    3106    3882#   3903    3964
STKQEN= 021771            3871#   3937    4048
```

J10


```
STKQIN  021756          3868#  3883*  3884   3935*  3936*  3937   3939*
STKQOU  021760          3869#  3884*  4046   4047*  4048   4050*
STKQSR  021762          3870#  3883   3939   4050
STKS    001160          1368#  3104*  3866   3888*  3919*  3921   3927*  3949   3965*  3975   3987*  4007*
STKSRV  022042          3885   3898#
STN   = 000002           958#   968   1814   1816#
STNPWR  023602          4225   4226   4246#
STPB    001166          1371#  3646*  3657
STPFLG  001173          1375#  3595   3657
STPS    001164          1370#  3644   3657
STRAP   023762          1828   4322#
STRAP2  024004          4333#  4344
STRP  = 000014          4337#  4346#  4347#  4348#  4349#  4350#  4351   4352#  4353   4354#  4355#  4356#  4357#
                        4358#
STRPAD  024016          4327   4344#
STSTM   001104          1136#
STSTNM  001116          1348#  3476   3508
STTYIN  023146          4061   4062   4074   4092   4106   4110#
STYPBN= ****** U        4350
STYPOS  021530          3807#  4349
STYPE   020552          3595#  3690   4337   4345
STYPEC  020764          3625   3632   3639   3644#  3645   4009
STYPEX  021032          3650   3652   3655#
STYPOC  021326          3747#  4346
STYPON  021342          3746   3749#  4348
STYPOS  021302          3742#  4347
SUNIT   001220          1396#  1445
SUNITM  001110          1138#
SUSWR   001232          1403#
$VECT1  001256          1428#  1882   1884
$VECT2  001260          1429#
$SGET4= 000000          2848#
$OFILL  021525          3743*  3747*  3757   3792#
$40CAT= ****** U        3486
.     = 040344          1095#  1099#  1107   1108#  1110#  1112#  1113#  1117#  1123   1124#  1126#  1128#  1140
                        1345#  1381   1487#  1490#  1493#  1496#  1499#  1823   1883   1886   1961   1975   2183
                        2223   2570   2597   2675   2709   2722   2856   3508   3657   3711#  3861#  3866   3870#
                        3871   3872#  4110#  4111   4118   4135   4159   4266#  5638#  5639#  5640#  5641#  5642#
                        5643#  5644#  5645#  5646
.$ASTA= ****** U        3663   3666
.$X   = 001100          1123#  1128
```

```
CKCHR      958#  2060  2130  3407  3421
CKDIG      958#  3142
CKNUM      958#
COMMEN    1078#
ENDCOM    1078#
ENDTYP    2774#  2782
ERRCAL    4359#  5160  5211  5214  5469  5530
ERRENT    3448#  3465
ERROR      972#  2193  2194  2195  2196  2197  2580  2581  2582  2583  2584  2585  2612  2613  2614
          2615  2616  2622  2629  2633  2635  2732  2733  2734  2735  2736  2739  2741  2745  5161
          5212  5216  5471  5531
ESCAPE    1078#  2191  2578  2610  2625  2730
GETPRI    1078#
GETSWR     958#  1078#  1864
LOOP       958#  2597  2722
MORETA    1339#  1436
MULT      1078#
NEWTST    1078#  1814
POP       1078#  3705  3706  3848  4144  4145  4306
PUSH      1078#  3666  3668  3689  3807  4125  4131  4286
REPORT    1078#
RM03.D       1#  4359
SCOPE      973#
SETPRI    1078#  4039
SETTRA    4337#  4346  4347  4348  4349  4351  4353  4354  4355  4356  4357
SETUP     1078#  1818
SKIP      1078#
SLASH     1078#
SPACE     1078#
STARS     1078#  1105  1120  1122  1129  1142  1146  1188  1192  1312  1316  1341  1381  1384  1805
          1809  1814  2776  2880  2884  3451  3580  3661  3719  3797  3865  3942  3957  4028  4052
          4121  4137  4164  4188  4207  4270  4316  4368  5756  5757  5843  5847
SWRSU     1078#  1834#
TRMTRP    4337#
TYPBIN    1078#
TYPDEC    1078#  2829  3548
TYPNAM    1078#
TYPNUM    1078#
TYPOCS    1078#  1905  2155
TYPOCT    1078#  3545  3970
TYPTXT    1078#
$SCMRE    1339#
$SCMTM    1339#
$SESCA    1078#
$SNEWT    1078#  1814
$SSET     4337#  4346  4347  4348  4349  4351  4353  4354  4355  4356  4357
$SSETM    1850#
$SSKIP    1078#
.EQUAT     958#   968
.HEADE     958#
.SETUP     958#  1811
.SWRHI     958#  1081
.SWRLO    1089#  1090  1091
.SACTI     958#  1103
.SAPTB     958#  1382#
.SAPTH     958#  1118
```

```
.$APTY    958#   3659
.$CATC    958#   1093
.$CMTA    958#   1339
.$D82D    958#   4205
.$EOP     958#   2774
.$ERRO    958#   3449
.$ERRT    958#
.$POWE    958#   4119
.$READ    958#   3863
.$SAVE    958#   4268
.$SB2D    958#   4186
.$SUPR    958#   4162
.$TRAP    958#   4314
.$TYPD    958#   3795
.$TYPE    958#   3578
.$TYPO    958#   3717

. ABS.  040344     000

 ERRORS DETECTED:  0

 DSKZ:DZRMAB,DSKZ:DZRMAB.SEQ/NL:MD:MC:CND:TOC/LI:ME/DOC/SOL/CRF=DSKM:RMDRV4.P11,DSKM:DZRMAB.P11
 RUN-TIME: 29 17 1 SECONDS
 RUN-TIME RATIO: 1024/48=20.9
 CORE USED:  38K  (75 PAGES)

 DOCUMENT PAGES:  127
```