RM05/3/2

RM05/3/2 PERF EXER
CZRMUAO

AH-S071A-MC
FICHE 1 OF 2

JUN 1980

MADE IN USA

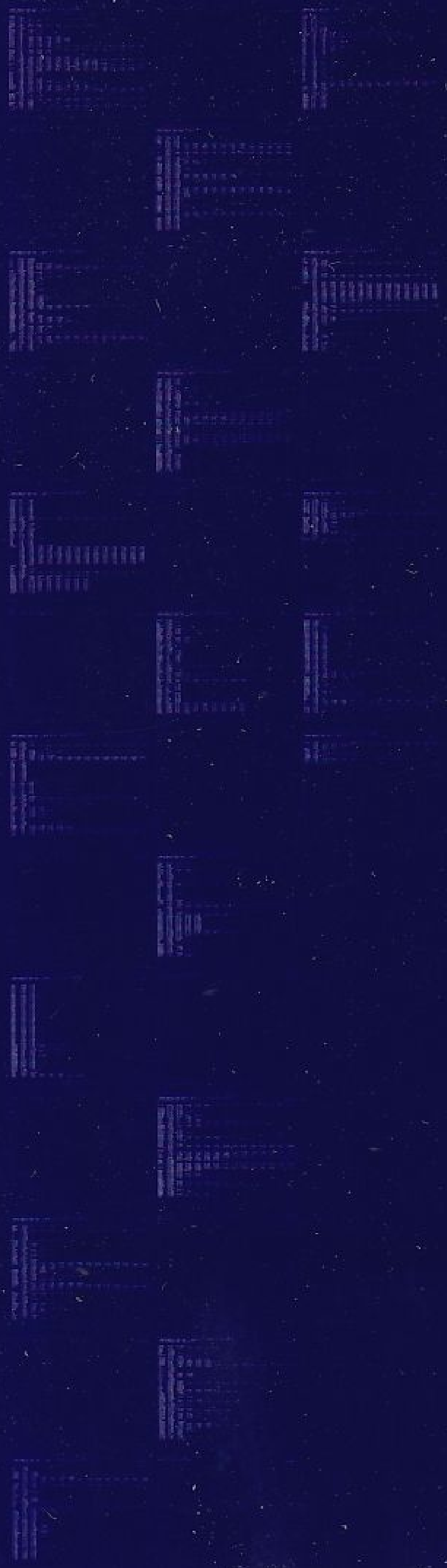
digital

RM05/3/2

RM05/3/2 PERF EXER
CZRMUAO

AH-S071A-MC
FICHE 2 OF 2

JUN 1980
COPYRIGHT© 1980
MADE IN USA



```
.REM    @

                             IDENTIFICATION
                             --------------

                PRODUCT CODE:       AC-S069A-MC

                PRODUCT NAME:       CZRMUA0 RM05/3/2 PERFORMANCE EXERCISER

                DATE CREATED:       APRIL 1980

                MAINTAINER:         CX DIAGNOSTIC GROUP

                AUTHOR:             MIKE LEAVITT




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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER
UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED
(WITH  INCLUSION OF DIGITAL'S COPYRIGHT NOTICE)  ONLY FOR USE IN SUCH
SYSTEM,  EXCEPT AS  MAY OTHERWISE  BE PROVIDED IN WRITING BY DIGITAL.

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

COPYRIGHT (C) 1980 DIGITAL EQUIPMENT CORPORATION
```

CONTENTS

1. ABSTRACT

THE RM05/3/2 PERFORMANCE EXERCISER PROGRAM IS DESIGNED TO PERFORM AN INTERACTIVE TEST ON RM DISK DRIVES CONNECTED TO A MASSBUS SUBSYSTEM. THE DRIVES MAY BE CONTROLLED BY EITHER AN RH11 OR AN RH70. IN ADDITION TO PERFORMING AN INTERACTIVE TEST OF THE DISK DRIVES ON THE SUBSYSTEM, THE PROGRAM IS INTENDED TO BE USED TO VERIFY THAT THE DRIVES UNDER TEST ARE PERFORMING TO THEIR DATA ERROR RATE AND SEEK ERROR RATE (SEE ERROR RATE SPECIFICATIONS).

THE PERFORMANCE EXERCISER PROGRAM WILL EXERCISE DRIVES CONNECTED AS EITHER SINGLE OR DUAL PORT UNITS. DUAL PORT DRIVES ARE TESTED BY LOADING AND RUNNING THE PROGRAM FROM BOTH CONTROLLING SYSTEMS. THE PROGRAM WILL EXERCISE A MIXED SYSTEM OF DUAL PORT AND SINGLE PORT DRIVES.

TO OBTAIN INTERACTIVE TESTING, OPERATIONS ON THE DRIVES ARE OVERLAPPED (OTHER DRIVES ARE PERFORMING SEEK/SEARCH OPERATIONS WHILE ONE DRIVE IS PERFORMING A DATA TRANSFER OR WRITE CHECK OPERATION). OPERATIONS AMONG THE DRIVES ARE OPTIMIZED SO THAT A HIGH SUBSYSTEM DATA TRANSFER RATE OR A HIGH POSITIONING OPERATION RATE IS MAINTAINED.

THE PERFORMANCE OF EACH DRIVE IS MONITORED BY THE PROGRAM. IF A DRIVE EXCEEDS A PRESET NUMBER OF ERRORS IN ANY OF SEVERAL CATEGORIES, THAT DRIVE IS AUTOMATICALLY DEASSIGNED. (THE OPERATOR MAY OVERRIDE THE AUTOMATIC DEASSIGNMENT FEATURE.) THE PROGRAM REPORTS PERFORMANCE STATISTICS FOR EACH DRIVE BEING EXERCISED ON REQUEST FROM THE OPERATOR OR AUTOMATICALLY AT AN INTERVAL DETERMINED BY THE OPERATOR.

ALL DATA TRANSFER COMMANDS ARE USED (I.E., WRITE DATA, WRITE HEADER & DATA, READ DATA, AND READ HEADER & DATA) AS WELL AS WRITE CHECK DATA AND WRITE CHECK HEADER & DATA COMMANDS. RECALIBRATE AND READ-IN PRESET COMMANDS ARE USED AT STARTUP AND DRIVE INITIALIZATION. RECALIBRATE, OFFSET, AND RETURN-TO-CENTERLINE COMMANDS ARE USED DURING ERROR RECOVERY.

THE DATA TRANSFER COMMANDS ARE SELECTED RANDOMLY EXCEPT FOR THE WRITE CHECK COMMANDS. THE WRITE CHECK COMMANDS ARE USED TO VERIFY A PREVIOUS WRITE OPERATION. THUS, WHEN A WRITE COMMAND IS SELECTED, THE DATA WRITTEN IS VERIFIED BY THE APPROPRIATE WRITE CHECK COMMAND.

DEPENDING UPON WHETHER PROGRAM HAS BEEN LOADED VIA APT AUTOMATIC MODE OR APT DUMP MODE WILL DETERMINE WHETHER; PROGRAM/OPERATOR COMMUNICATIONS ARE THROUGH THE KEYBOARD, DYNAMIC PROGRAM OPTIONS ARE SELECTED VIA SWITCH REGISTER SETTINGS AND ERRORS ARE REPORTED ON THE TELETYPE.

ALL COMMANDS, DATA PATTERNS, AND DATA BUFFER SIZES ARE SELECTED RANDOMLY BY THE PROGRAM. ADDITIONALLY THE ADDRESSES (EG, CYLINDER, TRACK, AND SECTOR) FOR EACH OPERATION ARE SELECTED RANDOMLY.

1.1 GENERAL NOTES

A. IN REFERENCE TO NUMBERS. TO INDICATE THE BASE OF A NUMBER

LARGER THAN SEVEN. A PERIOD(.) WILL FOLLOW THE NUMBER TO INDICATE DECIMAL OR NO PERIOD WILL FOLLOW THE NUMBER TO INDICATE OCTAL. IF THE NUMBER OCCURS AT THE END OF A SENTENCE, A DOUBLE PERIOD(..) INDICATES DECIMAL AND A SINGLE PERIOD(.) INDICATES OCTAL. ALSO, ANY REFERENCES TO TIME ARE ALWAYS IN DECIMAL.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
16K MEMORY (20K IF THE PROGRAM IS INCLUDED IN AN 'XXDP' CHAIN)
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

2.2 MEDIA

THE PERFORMANCE EXERCISER PROGRAM REQUIRES FORMATTED DISK PACKS GENERATED BY THE RM05/3/2 FORMATTER PROGRAM (CZRML-). THE PACKS MUST BE FORMATTED IN 32 SECTOR (16 BIT) MODE;THE ALTERNATE (30 SECTOR - 18 BIT) MODE IS NOT SUPPORTTED.

2.3 PRELIMINARY PROGRAMS

RM05/3/2 DISKLESS DIAGNOSTIC, PART 1 & 2

RM05/3/2 FUNCTIONAL TEST, PART 1, 2 & 3

3. OPERATING PROCEDURE

3.1 LOADING

THE PROGRAM MAY BE LOADED BY EITHER OF THE FOLLOWING MEDIA:

.PAPER TAPE, USING THE STANDARD PAPER TAPE PROCEDURE
.XXDP MEDIA, USING ANY XXDP DEVICE

3.2 STARTING

THE PROGRAM STARTS AT LOCATION 200

THE PROGRAM WRITES A DATA PATTERN TO ALL ON-LINE DRIVES IN A SEQUENTIAL SEEK MODE. UPON COMPLETION OF THE WRITE, THE PROGRAM GOES INTO A RANDOM TESTING MODE.

NOTE: PARAMETERS NOT INCLUDED IN THE TELETYPE DIAGLOGUE GROUP MUST BE CHANGED BEFORE THE PROGRAM IS STARTED.

3.3 START THE PROGRAM AT LOCATION 204 IF THE RH11 OR THE RH70 IS

NOT AT THE DESIRED ADDRESS. (DEFAULT IS 176700)

3.4 PROVIDED THE PROGRAM HAS BEEN LOADED AND STARTED VIA THE APT DUMP MODE OR THE DIAGNOSTIC IS RUNNING IN STAND ALONE PROCESSOR/DRIVE OPERATIONS ARE INITIATED AND CONTROLLED BY KEYBOARD COMMANDS AND SWITCH REGISTER SWITCH SETTINGS. IF THIS IS THE PROGRAM'S FIRST START, THE STATUS OF THE DRIVES ON THE SELECTED MASSBUS SUBSYSTEM WILL BE TYPED OUT. ON SUBSEQUENT STARTS, THIS TYPEOUT MAY BE INHIBITED BY SETTING SW<02>= 1.

HOWEVER, IF THE PROGRAM IS LOADED VIA APT SCRIPT MODE ALL SETUP AND SWITCH REGISTER SETTINGS WILL BE PROVIDED THROUGH THE APT E TABLE. TYPEOUTS FROM THE USER DIAGNOSTIC MAY OR MAYNOT BE INHIBITED DEPENDING UPON WHETHER OR NOT THE APPROPRIATE BIT IN THE E TABLE HAS BEEN SET.

3.5 PASS/TEST TERMINATION

A PASS IN RANDOM 'T' COMMAND MODE OR SEQUENTIAL 'T' COMMAND MODE IS DETERMINED BY EITHER BITS READ OR SEEKS PERFORMED. THE NUMBER OF BITS OR SEEKS REQUIRED FOR A PASS IS DERIVED FROM EITHER THE SOFT ERROR RATE SPECIFICATION OR THE SEEK ERROR RATE SPECIFICATION.

THE SPECIFICATIONS FOR THE RM DRIVE SPECIFY NO MORE THAN 1 SOFT ERROR (NON-PACK RELATED) IN 1 X 10^9 BITS READ OR NO MORE THAN 1 SEEK ERROR IN 1 X 10^6 SEEKS. THE NUMBER OF BITS OR SEEKS DETERMINING A PASS WERE SELECTED TO PROVIDE A 90% CONFIDENCE LEVEL THAT THE DRIVE IS PERFORMING TO THE APPLICABLE SPECIFICATION.

A PASS IN 'W' OR 'R' COMMAND MODE IS RELATED TO THE MAXIMUM DISK ADDRESS LIMITS SETUP BY THE USER.

3.5.1 PASS TERMINATION

END OF PASS FOR A SINGLE DRIVE IN THE RANDOM 'T' COMMAND MODE OR SEQUENTIAL 'T' COMMAND MODE, IS DETERMINED BY ONE OF THE FOLLOWING CONDITIONS.

A. IF PARAMETER 'ENDING' IS 1, END OF PASS OCCURS WHEN THE DRIVE HAS READ 1.875 X 10^8 WORDS (3 X 10^9 BITS).

B. IF PARAMETER 'ENDING' IS 0, END OF PASS OCCURS WHEN THE DRIVE HAS PERFORMED 3 X 10^6 SEEKS.

END OF PASS FOR A SINGLE DRIVE IN 'W' OR 'R' COMMAND MODE, IS DETERMINED AS FOLLOWS.

A. WHEN A SEQUENTIAL SEEK IS MADE BEYOND THE MAXIUM DISK ADDRESS LIMITS SET BY THE USER, THE PASS IS CONSIDERED ENDED.

3.5.2 TEST TERMINATION

IF SW04 IS CLEAR, THE TEST FOR A DRIVE IS TERMINATED WHEN:

A. THE DRIVE HAS COMPLETED THE NUMBER OF PASSES SPECIFIED IN PARAMETER 'PASSES'.

B. THE TOTAL ERRORS ACCUMULATED EXCEED 100..

```
                C.  A FATAL ERROR OCCURS: EM12 OR EM14.
                D.  OPERATOR DEASSIGNS THE DRIVE

        3.6     RUN TIME

                THE EXERCISER PROGRAM MAY BE RUN IN TWO MODES.  THE MODE IS
                DETERMINED BY THE VALUE IN PARAMETER 'SIZE'.  IF 'SIZE' IS ONE
                SECTOR, THE PROGRAM RUNS IN A SEEK HEAVY MODE; IF 'SIZE' APPROACHES
                1/2 TRACK IN SIZE (4096. WORDS) THE PROGRAM RUNS IN A DATA TRANSFER
                HEAVY MODE.  THE PROGRAM RUN TIME VARIES GREATLY DEPENDING ON
                THE OPERATION MODE SELECTED, THE MEMORY AVAILABLE OVER 16K, THE
                READ/WRITE RATIO PARAMETER - 'RATIO', AND BY SWITCHES 0, 1, & 2.

        3.6.1   DATA TRANSFER MODE

                1 DRIVE - APPROX. 3.6 HRS (TO REACH 1.875 X 10^8 WORDS)

                  TO

                8 DRIVES - APPROX. 16 HRS (FOR ALL DRIVES TO REACH 1.875 X 10^8 WORDS)

                NOTE:   IF THE PROGRAM IS RUN WITH BOTH SW<00> AND SW<01>
                        SET, THE RUN TIMES SHOULD BE ABOUT 20% FASTER.

        3.6.2   SEEK VERIFICATION MODE

                PARAMETER 'SIZE' = 1 SECTOR (256 WORDS)
                PARAMETER 'MAXTRK' = 'MINTRK'
                PARAMETER 'MAXSEC' = 'MINSEC'
                SW<00> = 1 (READ ONLY MODE)

                1 DRIVE - APPROXIMATELY 25 HRS (3 X 10^6 SEEKS)

                  TO

                8 DRIVES - APPROXIMATELY 40 HRS (3 X 10^6 SEEKS FOR ALL DRIVES)

        3.7     UNIBUS & VECTOR ADDRESSES

                THE PROGRAM ASSUMES THE FOLLOWING UNIBUS AND VECTOR ADDRESSES.  (REFER
                TO SECTION 4.2.2 FOR THE LOCATIONS AT WHICH TO CHANGE THESE ADDRESSES.)

                                        UNIBUS          VECTOR
                        UNIT            ADDRESS         ADDRESS
                        ----            -------         --------

                        RH11 OR RH70    176700            254

                        TTY PRINTER     177564          NOT USED

                        TTY KEYBOARD    177560            60

                        KW11-L          177546           100

                        KW11-P          172542           104
```

```
3.8     DUAL PORT OPERATION

        A.  LOAD THE PERFORMANCE EXERCISER PROGRAM INTO BOTH PROCESSORS.

        B.  SWITCH THE  'CONTROLLER SELECT'  SWITCH TO  'A/B'  ON EACH DRIVE
             WHICH IS TO BE TESTED AS A DUAL PORT DRIVE AND CYCLE THE DRIVES UP.

        C.  START THE PROGRAM IN EACH PROCESSOR.  RUN THE PROGRAM AS THOUGH
             EACH PROCESSOR WERE RUNNING INDEPENDENTLY OF THE OTHER.

3.9     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

3.10    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:
                        NOT USED

                13. CONTROLLER DESCRIPTOR WORDS:
                        NOT USED

                14. CONTROLLER DESCRIPTOR WORDS:
                        NOT USED




        4.      CONTROLLING THE PROGRAM
                -----------------------

                THE FOLLOWING KEYBOARD CONVENTIONS ARE USED BY THE KEYBOARD ENTRY
                ROUTINES IN THE PROGRAM:

                A.  TO DELETE AN INCORRECT CHARACTER FROM AN ENTRY STRING, TYPE A
                    'RUBOUT'. TYPING A RUBOUT WILL DELETE SUCESSIVE CHARACTERS
                    FROM THE INPUT.

                B.  TO DELETE AN ENTIRE LINE, TYPE A 'CONTROL U' (^U).

                C.  AN ENTRY MUST BE TERMINATED BY EITHER A 'CARRIAGE RETURN' OR
                    A 'PERIOD'.  THE 'PERIOD' TERMINATION IS RECOGNIZED BY THE
                    PROGRAM AS A DEFAULT ENTRY REQUEST.  WHEN A LINE IS TERMINATED
                    BY A 'PERIOD' INSTEAD OF A 'CARRIAGE RETURN', THE PROGRAM WILL
                    ACCEPT THE ENTERED VALUE AND WILL DEFAULT TO THE PRELOADED VALUES
                    FOR ANY REMAINING ENTRIES.

                D.  IF A 'CONTROL C' IS TYPED DURING KEYBOARD ENTRY, THE PROGRAM
                    WILL RETURN TO THE BEGINNING OF THE GROUP BEING ENTERED.
        4.1     DATE & OPERATOR IDENTIFICATION
```

ASSUMING THE DIAGNOSTIC HAS BEEN LOADED BY ANY OTHER MODE OTHER THAN THE APT SCRIPT MODE THE PROGRAM WHEN IT IS INITIALLY STARTED, WILL ASK FOR DATE AND OPERATOR I.D. ENTRIES. (THE REQUEST FOR THESE ENTRIES OCCURS ONLY WHEN THE PROGRAM IS FIRST STARTED AND WILL NOT APPEAR WHEN THE PROGRAM IS RESTARTED.) THESE ENTRIES ARE OPTIONAL AND MAY BE BYPASSED BY ENTERING A 'CARRIAGE RETURN' IN RESPONSE TO THE REQUEST. THE PROGRAM DOES NOT EDIT OR CHECK EITHER ENTRY. UP TO 8 CHARACTERS OF DATE INFORMATION AND UP TO 6 CHARACTERS OF OPERATOR IDENTIFICATION MAY BE ENTERED. BOTH THE DATE AND THE OPERATOR I.D. WILL BE TYPED WHEN THE 'SA' COMMAND IS PREFORMED (SEE SECTION 4.4.3).

4.2 PARAMETERS

WHEN THE PROGRAM IS STARTED FROM LOCATION 204, THE OPERATOR WILL BE ASKED TO ENTER PARAMETERS. THE FOLLOWING MESSAGE WILL BE DISPLAYED:

```
          CHANGE PARAMETERS ?
```

THE OPERATOR MUST ENTER A 'Y' IF PARAMETER ENTRIES ARE TO BE MADE. ANY OTHER CHARACTER IS ACCEPTED AS A 'NO' ENTRY. THE PROGRAM WILL IDENTIFY THE PARAMETER BY THE NAME GIVEN BELOW, DISPLAY THE CURRENT VALUE OF THE PARAMETER AND WAIT FOR THE ENTRY. THE PROGRAM WILL TYPE 'INVALID ENTRY' IF THE ENTRY IS NOT CORRECT AND WAIT FOR A CORRECT ENTRY TO BE TYPED.

NOTE: WHEN THE DIAGNOSTIC IS LOADED VIA APT SCRIPT MODE ALL PARAMETERS ARE LOADED FROM THE APT SYSTEM E TABLE. THIS INCLUDES THE SOFTWARE SWITCH REGISTER. THEREFORE A WORST CASE CONDITION WILL BE SET IN THE E TABLE FOR NORMAL AUTOMATIC OPERATION.

4.2.1 KEYBOARD ENTRY PARAMETERS

NAME	BASE	DEFAULT VALUE	VALUE RANGE	FUNCTION
SIZE	10	(SEE NOTE)		CONTROLS THE MAXIMUM BUFFER SIZE USED FOR DATA TRANSFERS
PASSES	10	1	1 - 999.	NUMBER OF PASSES TO END OF TEST.
MINUTE	10	120	0 - 256.	DETERMINES THE INTERVAL (IN MINUTES) BETWEEN AUTOMATIC PERFORMANCE SUMMARY TYPEOUTS ERRORS PRINTED OUT IF SW<07>=0

RANDOM	8	000000	0 OR 1	IF PARAMETER = 0, THE DATA TRANSFER WORD COUNT IS RANDOMLY SELECTED BETWEEN 4 AND THE VALUE IN 'SIZE'.
				IF PARAMETER = 1, THE DATA TRANSFER WORD COUNT WILL BE THE VALUE IN 'MAXDL'

ENDING	8	000001	0 OR 1	IF PARAMETER = 1, END OF PASS DETERMINED BY THE 'WORDS READ' COUNT.

```
                                                        IF PARAMETER = 0, END OF PASS
                                                        IS DETERMINED BY THE NUMBER
                                                        OF SEEKS.
                    FORMAT    8      000001     0 OR 1  IF PARAMETER = 0; DO NOT
                                                        PERFORM WRITE HEADER & DATA
                                                        ORDERS; IF PARAMETER > 0,
                                                        PERFORM WRITE HEADER & DATA
                                                        ORDERS
                    RATIO     8      000003     0 - 7   CONTROLS THE APPROXIMATE
                                                        RATIO OF READ TO WRITE
                                                        ORDERS.

                                                        VALUE   R/W RATIO

                                                          0       15/1
                                                          1       7/1
                                                          2       6/2
                                                          3       5/3
                                                          4       4/4
                                                          5       3/5
                                                          6       2/6
                                                          7       1/7
                    MESSAGE   8      000001     0 OR 1  IF PARAMETER = 1, DO NOT PRINT
                                                        ERROR MESSAGES FOR DATA ERRORS
                                                        OCCURING AT LOCATIONS DEFINED
                                                        BY THE OPERATOR AS BAD PACK
                                                        LOCATION.
                                                        IF PARAMETER = 0, PRINT ERROR
                                                        MESSAGES ASSOCIATED WITH BAD
                                                        PACK LOCATIONS.
                    PATTERN 10       000000     0 - 15. IF PARAMETER=0,DATA
                                                        PATTERN IS RANDOMLY
                                                        SELECTED.
                                                        IF PARAMETER>0,SPECIFIES
                                                        ONE OF THE 15 PATTERNS.
                                                        THE SELECTED DATA PATTERN
                                                        IS POINTED BY THE PARAMETER
                                                        'PATTERN'.
                    HEADER  8        000001     0 OR 1  IF PARAMETER=0,RANDOM
                                                        DATA BLOCK ADDRESS IS
                                                        USED IN 'T' COMMAND
                                                        IF PARAMETER=1,SEQUENTIAL
                                                        DATA BLOCK IS USED IN
                                                        'T' COMMAND.

                    NOTE:  THE PROGRAM WILL SELECT A MAXIMUM BUFFER SIZE WHICH
                           IS DETERMINED BY THE MEMORY AVAILABLE.  THE MAXIMUM BUFFER
                           SIZE ASSIGNED BY THE PROGRAM IS 8192.(ONE TRACK) WORDS.  THE
                           OPERATOR MAY SPECIFIY ANY OTHER MAXIMUM SIZE AS LONG AS THE
                           VALUE SPECIFIED IS AT LEAST 4 WORDS BUT NO LARGER THAN
                           THE INITIAL VALUE OF 'MAXDL' DETERMINED BY THE PROGRAM.


            4.2.2   PERIPHERAL ADDRESSES AND OTHER LOCATIONS OF INTEREST

                    TO ALTER THESE LOCATIONS, THE OPERATOR MUST MAKE MANUAL ENTRIES
                    BEFORE THE PROGRAM IS STARTED.  THE KEYBOARD ENTRY ROUTINE DOES
```

```
                  NOT PROVIDE ACCESS TO THESE LOCATIONS.

                  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
                  1274      $LKCSR    172540             ADDRESS OF KW11-P STATUS REGISTER
                  1276      $LKCSB    172542             ADDRESS OF KW11-P COUNTER BUFFER
                  1300      $LPVEC       104             KW11-P VECTOR ADDRESS
                  1302      $LKS      177546             ADDRESS OF KW11-L STATUS REGISTER
                  1304      $LLVEC       100             KW11-L VECTOR ADDRESS
                  1312      HZ            74             74 (60 DECIMAL) IF SYSTEM IS 60 HZ;
                                                         62 (50 DECIMAL) IF SYSTEM IS 50 HZ.

                  THE RH11-RH70 ADDRESS AND VECTOR MAY BE CHANGED WHEN THE PROGRAM
                  IS STARTED FROM LOCATION 204 OR IF THE PROGRAM DOES NOT RECEIVE
                  A RESPONSE WHEN IT ACCESSES THE DEFAULT RH11-RH70 ADDRESS.

          4.2.3   PARAMETERS FOR THE FIRST OPERATION

                  THE FOLLOWING PARAMETERS ARE USED FOR THE INITIAL OPERATION (IN
                  ADDITION TO THE 'MINIMUM' ADDRESS VALUES).

                                      INITIAL    VALUE
                  LOC       TAG        VALUE     RANGE                   FUNCTION
                  ---       ---       -------    -----                   --------

                  1514      BEGPAT      10        1 - 15         THE CODE FOR THE STARTING
                                                                 PATTERN.  (IF A WRITE ORDER
                                                                 OR A WRITE CHECK ORDER IS
                                                                 SPECIFIED IN 'BEGCOD')
                  1516      BEGCOD       5       0 - 5           THE INITIAL COMMAND FOR EACH DRIVE
                                                                 EXERCISED.
                                                                 0 = WRITE CHECK DATA
                                                                 1 = WRITE CHECK HEADER & DATA
                                                                 2 = WRITE DATA
                                                                 3 = WRITE HEADER & DATA
                                                                 4 = READ DATA
                                                                 5 = READ HEADER & DATA
                  1520      BEGSIZ     402       2 - SIZE        THE BUFFER SIZE FOR THE FIRST
                                                                 DATA TRANSFER OPERATION.

          4.3     SWITCH REGISTER SETTINGS

                  SW <15> = 1         HALT ON ERROR
                  SW <13> = 1         INHIBIT ERROR TYPEOUT
                  SW <10> = 1         RING THE TELETYPE BELL IF ERROR
                  SW <7>  = 1         DISPLAY ALL DATA COMPARE ERRORS
                  SW <6>  = 1         DO NOT ALTER THE CURRENT OPERATION PARAMETERS
                  SW <5>  = 1         PARTIAL REGISTER DISPLAY IF ERROR; DO NOT DISPLAY
                                      ECC CORRECTION RESULTS
                  SW <4>  = 1         INHIBIT MAXIMUM ERROR COUNT CHECK; DO NOT DEASSIGN
                                      DRIVES WHEN END OF TEST IS REACHED.
                  SW <3>  = 1         DISPLAY THE SECTOR IN ERROR (BEFORE RETRY ATTEMPTS)
```

```
                IF 'DCK', 'DTE', OR 'WCF' ERRORS OR AFTER THE 28TH
                RETRY IF UNCORRECTABLE 'DCK' ERROR.
                IF DATA COMPARE ERRORS & SW<7> SET, DISPLAY REST
                OF BUFFER
SW <2>  = 1     INHIBIT SUBSYSTEM STATUS TYPEOUT DURING STARTUP.
                INHIBIT PERFORMANCE REPORT AFTER SPECIFIED TIME
SW <1>  = 1     INHIBIT DATA COMPARSION AFTER READ ORDERS
SW <0>  = 1     READ ONLY MODE
```

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. 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 IN KEYBOARD ENTRY MODE, OR IS AT A HIGHER PRIORITY PROCESSING AN DRIVE 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 SETTINGS 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.

4.4 KEYBOARD COMMANDS

NOTE: ALL KEYBOARD COMMANDS WILL BE DISABLED IF DIAGNOSTIC IS RUNNING IN THE APT SCRIPT MODE.

THROUGH THE KEYBOARD COMMANDS, THE OPERATOR MAY ASSIGN DRIVES FOR TEST ('T' COMMAND), WRITE AND CHECK DATA PACKS ('W' COMMAND), PERFORM WRITE DATA AND FOLLOWED BY TEST ('WT' COMMAND), PERFORM A SEQUENTIAL READ OF A PACK ('R' COMMAND), REQUEST A DRIVE PERFORMANCE SUMMARY ('S' COMMAND), OR DEASSIGN A DRIVE WHICH IS BEING TESTED, READING, OR WRITING ('D' COMMAND).

IF THE START ADDRESS WAS 204, THE FOLLOWING MESSAGE WILL BE TYPED AFTER THE PROGRAM HAS BEEN INITIALIZED.

```
        'PROGRAM INITIALIZATION COMPLETE
        'TYPE A CONTROL C TO ENTER COMMANDS'
```

KEYBOARD ENTRIES WILL NOT BE RECOGNIZED UNTIL THE OPERATOR TYPES A 'CONTROL C'. WHEN THE PROGRAM SEES A 'CONTROL C' ENTRY, IT WILL SUSPEND THE SCHEDULING OF FURTHER DEVICE OPERATIONS AND WAIT UNTIL ALL OUTSTANDING ORDERS HAVE TERMINATED. THE PROGRAM WILL ENTER COMMAND ENTRY MODE AND TYPE THE FOLLOWING PROMPTING MESSAGE:

```

        'HH:MM:SS
        'ENTER COMMANDS:'

THE PROGRAM WILL THEN ACCEPT ANY OF THE VALID COMMANDS.  AT THE
COMPLETION OF A COMMAND, THE PROGRAM WILL EXIT COMMAND MODE; THE
OPERATOR MUST TYPE ANOTHER 'CONTROL C' TO RETURN THE PROGRAM TO
COMMAND MODE.  IF THE COMMAND ENTERED SPECIFIED AN 'A'
DRIVE NUMBER, THE PROGRAM WILL REMAIN IN COMMAND MODE UNTIL ALL
AVAILABLE DRIVES HAVE BEEN PROCESSED.

THE 'WT', 'T', 'W' AND 'R' COMMANDS REQUIRE DRIVE I.D, ADDRESS LIMITS
AND BAD LOCATION ADDRESS ENTRIES FOR THE DRIVE BEING REFERENCED.

THE PROGRAM WILL FIRST TELL THE USER WHICH DRIVE IS BEING REFERENCED
FOR CHANGES.

        *********** DRIVE # N

THE PROGRAM WILL THEN ASK FOR A DRIVE IDENTIFICATION NUMBER WITH THE
FOLLOWING TYPEOUT:

        'ENTER DRIVE I.D.:'

THE OPERATOR MAY ENTER AN I.D. NUMBER FOR THE DRIVE OF UP TO 6
CHARACTERS IN LENGTH.  THIS I.D. WILL BE DISPLAYED, ALONG WITH THE
DATE AND OPERATOR I.D. ENTRIES (SEE SECTION 4.1), WHEN THE 'SA'
COMMAND IS EXECUTED.  THE OPERATOR MAY ENTER ANY CHARACTER STRING,
TERMINATED BY A 'CARRIAGE RETURN', OR A 'PERIOD' ONLY (NULL ENTRY)
IN RESPONSE TO THE I.D. REQUEST.

THE PROGRAM WILL THEN ASK FOR ADDRESS LIMIT CHANGES WITH THE FOLLOWING
TYPEOUT:

        'ENTER ADDRESS LIMITS:'

THE PROGRAM WILL REQUEST VALUES FOR THE FOLLOWING ADDRESS LIMIT
PARAMETERS.

        DEFAULT   VALUE
NAME     VALUE    RANGE                   FUNCTION
----    -------   -----                   --------
MINCYL     0      0 - 822.       THE MINIMUM CYLINDER ADDRESS
MAXCYL   822.     0 - 822.       THE MAXIMUM CYLINDER ADDRESS
MINTRK     0      0 - NN         THE MINIMUM TRACK ADDRESS
MAXTRK    NN      0 - NN         THE MAXIMUM TRACK ADDRESS
MINSEC     0      0 - 31.        THE MINIMUM SECTOR ADDRESS
MAXSEC   31.      0 - 31.        THE MAXIMUM SECTOR ADDRESS

WHERE 'NN' IS 4. FOR AN RM03/02 AND 18. FOR AN RM05.

NOTE: WHEN THE 'T' COMMAND IS SELECTED, THE MINIMUM CYLINDER,
      TRACK, OR SECTOR ADDRESS MAY BE SPECIFIED AS BEING LARGER
      THAN THE MAXIMUM ADDRESS. WHEN THESE VALUES ARE INVERTED,
      THE PROGRAM WILL SELECT ADDRESSES BETWEEN THE 'MAX' ADDRESS
      AND THE UPPER PHYSICAL LIMIT.

```

THE PROGRAM WILL THEN ASK FOR BAD SPOT ADDRESSES WITH THE FOLLOWING TYPEOUTS:

```
        'ENTER BAD SPOT ADDRESSES:'
```

THIS ROUTINE ALLOWS THE USER TO MANUALLY ENTER THE BAD SECTORS INTO A BAD SECTOP TABLE FOR THAT PARTICULAR DRIVE.

THE PROMPT USED TO ENTER THE CYLINDER ADDRESS IS AS FOLLOWS:

```
        'CYLNDR   -1 / '
```

TO ENTER A BAD CYLINDER ADDRESS INTO THE BAD SPOT TABLE, TYPE ANY NUMBER FROM 0 - 822., FOLLOWED BY A <CR>. HOWEVER, TO ENTER ALL TRACKS/SECTORS ON THE CURRENT CYLINDER AS BEING BAD AND EXIT THE ROUTINE, JUST TYPE A NUMBER AS ABOVE, FOLLOWED BY PERIOD(.) AND <CR>. TO EXIT WITH NO ENTRY, TYPE A <CR> OR TYPE A PERIOD(.) FOLLOWED BY A <CR>.

THE PROMPT USED TO ENTER THE TRACK ADDRESS IS AS FOLLOWS:

```
        'TRACK   -1 / '
```

TO ENTER THE BAD TRACK ADDRESS INTO THE BAD SPOT TABLE, TYPE ANY NUMBER FROM 0 - NN, FOLLOWED BY A <CR>. TO ENTER ALL TRACKS ON THE PREVIOUSLY ENTERED CYLINDER AS BEING BAD, JUST TYPE A <CR> WITH NO ENTRY, WHICH WILL BRING YOU TO THE NEXT PROMPT OR JUST TYPE A PERIOD(.) FOLLOWED BY A <CR>, WHICH WILL CAUSE YOU TO EXIT THE ROUTINE.

WHERE 'NN' IS 4 FOR AN RM03/02 OR 18. FOR AN RM05.

THE PROMPT USED TO ENTER THE SECTOR ADDRESS IS AS FOLLOWS:

```
        'SECTOR  -1 / '
```

TO ENTER THE BAD SECTOR ADDRESS INTO THE BAD SPOT TABLE, TYPE ANY NUMBER FROM 0 - 31., FOLLOWED BY A <CR>. TO ENTER ALL SECTORS ON THE PREVIOUSLY ENTERED CYLINDER/TRACK AS BEING BAD, JUST TYPE A <CR> WITH NO ENTRY, WHICH WILL BRING YOU TO THE NEXT CYLINDER PROMPT OR JUST TYPE A PERIOD(.) FOLLOWED BY A <CR>, WHICH WILL CAUSE YOU TO EXIT THE ROUTINE.

FOR MORE INFORMATION (SEE SECTIONS 6.3 AND 6.4)

EXAMPLE #1

```
        CYLNDR   -1 / <CR>        ;NO ENTRIES AND EXIT ROUTINE
```

EXAMPLE #2

```
        CYLNDR   -1 / 820<CR>     ;ENTERED 820. AS BAD CYLINDER.
        TRACK    -1 / 3<CR>       ;ENTERED 3 AS BAD TRACK.
        SECTOR   -1 / 20.<CR>     ;ENTERED 20. AS BAD SECTOR AND
                                  ;EXIT ROUTINE
```

```

                EXAMPLE #3

                        CYLNDR   -1 / 820.<CR>  ;ENTERED 820. AS BAD CYLINDER.
                                                ;INDICATE ALL TRACKS AND SECTORS
                                                ;ON THAT CYLINDER AS BEING BAD.
                                                ;EXIT ROUTINE
                EXAMPLE #4

                        CYLNDR   -1 / 820<CR>   ;SAME AS EXAMPLE #3
                        TRACK    -1 / .<CR>

                EXAMPLE #5

                        CYLNDR   -1 / 820<CR>   ;SAME AS EXAMPLE #3
                        TRACK    -1 / <CR>
                        SECTOR   -1 / .<CR>

                EXAMPLE #6

                        CYLNDR   -1 / 820<CR>   ;ENTERED 820. AS BAD CYLINDER.
                        TRACK    -1 / <CR>      ;INDICATE ALL TRACKS AS BEING BAD.
                        SECTOR   -1 / 20<CR>    ;ENTERED 20. AS BAD SECTOR.
                        CYLNDR   -1 / <CR>      ;EXIT ROUTINE

                EXAMPLE #7

                        CYLNDR   -1 / 820<CR>   ;ENTERED 820. AS BAD CYLINDER.
                        TRACK    -1 / 3<CR>     ;ENTERED 3 AS BAD TRACK.
                        SECTOR   -1 / .<CR>     ;INDICATE ALL SECTORS OF THE ABOVE
                                                ;CYLINDER/TRACK AS BEING BAD AND
                                                ;EXIT ROUTINE

        4.4.1   'T' COMMAND

                USED TO ASSIGN A DRIVE(S) FOR A RANDOM TEST. THIS COMMAND IS REQUIRED
                TO PERFORM THE TEST OF THE DRIVE(S).  THE OTHER COMMANDS ARE CONVIENCE
                COMMANDS OR SUPPORT COMMANDS.

                FORMAT: TN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINIATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  T0<CR> - ASSIGN DRIVE 0 FOR TEST
                          TA<CR> - ASSIGN ALL AVAILABLE DRIVES FOR TEST

                NOTE:  DRIVE OPERATION BEGINS IMMEDIATELY AFTER COMMAND IS ENTERED.


        4.4.2   'D' COMMAND

                USED TO DEASSIGN A DRIVE(S) BEING EXERCISED.

                FORMAT:  DN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINIATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  D0<CR> - DEASSIGN DRIVE 0
                          DA<CR> - DEASSIGN ALL DRIVES BEING TESTED.

                NOTES:  1. IF THE 'D' COMMAND REFERENCES A
                           DRIVE NOT ASSIGNED THE PROGRAM
                           WILL TYPEOUT '?DRIVE NOT ASSIGNED'

                        2. THE DRIVES WILL BE DEASSIGNED AS THEIR
                           OPERATIONS COMPLETE.

                        3. IF 'DA' IS USED,ALL DRIVES BEING TESTED
                           WILL BE DEASSIGNED; THE MESSAGE IN (1) WILL
                           BE DISPLAYED FOR ALL DRIVES NOT BEING TESTED.


        4.4.3   'S' COMMAND

                USED TO REQUEST A PERFORMANCE SUMMARY TYPEOUT FOR THE REFERENCED
                DRIVE(S).

                FORMAT:  SN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINIATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  S0<CR> - TYPEOUT PERFORMANCE SUMMARY FOR DRIVE 0
                          SA<CR> - TYPEOUT PERFORMANCE SUMMARY FOR ALL DRIVES
                                   BEING TESTED.

                NOTES:  1.  IF PARAMETER 'MINUTE' IS NOT ZERO, THE PROGRAM
                            WILL AUTOMATICALLY DISPLAY A PERFORMANCE SUMMARY
                            FOR EACH DRIVE BEING TESTED AT A RATE DETERMINED BY
                            'MINUTE'.

                        2.  IF THE 'SA' COMMAND IS USED, THE PROGRAM WILL TYPEOUT
                            THE OPERATOR ENTERED DATE, OPERATOR I.D., AND THE DRIVE
                            I.D. FOR EACH DRIVE BEING TESTED.  THE DATE AND OPERATOR
                            I.D. WILL NOT BE TYPED OUT IF NO DRIVES ARE BEING TESTED.
        4.4.4   'W' COMMAND

                USED TO WRITE A DATA PACK WITH DATA ACCEPTABLE TO THE PERFORMANCE
                 EXERCISER PROGRAM.

                FORMAT: WN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  W0<CR> - GENERATE A DATA PACK ON DRIVE 0.
                          WA<CR> - WRITE DATA PACKS ON ALL AVAILABLE DRIVES.

                NOTES:  1.  DATA PATTERNS GENERATED BY THE FORMATTER PROGRAM
```

(CZRML-) ARE COMPATIBLE.

2. THE 'W' COMMAND MUST BE USED TO WRITE A NEW PACK UNDER TEST BEFORE OTHER COMMANDS ARE ISSUED; IF THE DISK PACK HAS BEEN WRITTEN BY ANY PROGRAM OTHER THAN THE FORMATTER.

3. THE 'W' COMMAND PERFORMS A SEQUENTIAL WRITE OF THE PACK USING A 'WRITE DATA' COMMAND. THE DATA PATTERN USED FOR EACH WRITE IS SELECTED RANDOMLY. HOWEVER, THE OPERATION OF THE COMMAND IS SEQUENTIAL, BEGINNING AT 'MINCYL', 'MINTRK' AND CONTINUING TO 'MAXCYL', 'MAXTRK'.

4. THE 'W' COMMAND DOES NOT WRITE HEADERS AND ASSUMES THAT THE FORMAT OF THE PACK IS GOOD.

5. THE 'W' COMMAND CANNOT BE STARTED IF SWITCH 0 (READ ONLY MODE) IS SET. IF SWITCH 0 IS SET DURING THE OPERATION OF THE 'W' COMMAND, THE DRIVE PERFORMING THE 'W' COMMAND WILL IGNORE THE SWITCH.

4.4.5 'R' COMMAND

USED TO PERFORM A SEQUENTIAL READ OF THE PACK.

FORMAT: RN<CR>

N = DRIVE NUMBER. MAY BE 0 TO 7 OR 'A'. ENTRY MUST BE TERMINATED BY A CARRIAGE RETURN <CR>.

EXAMPLE: R0<CR> - READ THE PACK ON DRIVE 0.
RA<CR> - READ THE PACKS ON ALL OF THE ONLINE DRIVES.

NOTES: 1. THE PROGRAM WILL PERFORM A NORMAL CHECK OF ALL DATA READ. HOWEVER, ALL OPERATIONS WILL BE SEQUENTIAL.

2. THE PROGRAM WILL READ THE PACK STARTING AT THE ADDRESS SPECIFIED BY 'MINCYL', 'MINTRK' TO THE ADDRESS SPECIFIED BY 'MAXCYL', 'MAXTRK'. THE READ WILL BE SEQUENTIAL.

4.4.6 'WT' COMMAND

USED TO PERFORM A SEQUENTIAL WRITE PACK AND FOLLOWED BY A 'T' COMMAND.

FORMAT: WTN<CR>

N = DRIVE NUMBER 0 TO 7 OR ''A''. ENTRY MUST BE TERMINATED BY A CARRIAGE RETURN <CR>.

EXAMPLE: WT0<CR> - WRITE PACK 0 AND TEST PACK 0
WTA<CR> - WRITE ALL PACKS AND TEST ALL PACKS

```
4.4.7   GENERAL COMMAND INFORMATION

        A.    CONTROL-C MUST BE ENTERED BEFORE ISSUING ANY COMMAND.

        B.    T, R, W AND WT COMMANDS ARE EXCLUSIVE TO ONE ANOTHER
              ON THE SAME DRIVE UNDER TESTING.
              D COMMAND MUST BE ENTERED IN ORDER TO SWITCH COMMANDS
              AMONG T, R, W AND WT.

        C.    S COMMAND CAN BE ENTERED ANY TIME DURING THE TEST.

        D.  THE ERROR RESPONSES FROM THE PROGRAM ARE AS FOLLOWS

              RESPONSE                            COMMAND(S)
              --------                            ----------

            ?DRIVE N IS LOAD DEVICE               T, W, R, WT
            ?DRIVE N OFFLINE                      T, W, R, WT
            ?DRIVE N NOT ASSIGNED                 D, S
            ?DRIVE N ALREADY ASSIGNED             T, W, R, WT
            ?DRIVE N NOT PRESENT                  T, W, R, WT
            ?DRIVE N UNSAFE                       T, W, R, WT
            ?DRIVE N NOT AN RM05/3/2              T, W, R, WT


5.      PERFORMANCE SUMMARY TYPEOUT
        ---------------------------

5.1     THE PROGRAM WILL DISPLAY A PERFORMANCE SUMMARY FOR THE DRIVES
        BEING EXERCISED.  THIS SUMMARY WILL BE DISPLAYED AUTOMATICALLY
        IF THE PARAMETER 'INTRVL' IS NOT ZERO OR CAN BE DISPLAYED ON
        REQUEST BY THE OPERATOR THROUGH THE USE OF THE 'S' COMMAND.
        THE PERFORMANCE SUMMARY TYPEOUT CONTAINS THE FOLLOWING FIELDS:

        'DRV'          THE DRIVE NUMBER
        'PASS'         THE PRESENT PASS COUNT FOR THE DRIVE
        'ORDERS'       THE NUMBER OF ORDERS PERFORMED BY THE DRIVE
        'SEEKS'        THE NUMBER OF SEEK OPERATIONS THE DRIVE PERFORMED
        'WRDS XFER'    THE TOTAL NUMBER OF WORDS WRITTEN AND READ BY THE DRIVE
        'WRDS READ'    THE TOTAL NUMBER OF WORDS READ BY THE DRIVE
        'SOFT'         THE NUMBER OF SOFT DATA ERRORS
        'HARD'         THE NUMBER OF HARD DATA ERRORS
        'SKI'          THE NUMBER OF 'SKI' ERRORS
        'MISP'         THE NUMBER OF POSITIONING ERRORS
        'OTHER'        THE TOTAL ERRORS OF OTHER TYPES

        NOTE:  ERRORS EM1, EM2, EM3, EM4, EM5, & EM10 ARE NOT INCLUDED IN
               THE 'OTHER' ERROR TOTAL.

5.2     SOFT/HARD ERROR DEFINITIONS

5.2.1   HARD ERRORS

        A.  A 'DTE' (DRIVE TIMING ERROR) OR A 'DCK' (DATA CHECK ERROR)
```

```
                                WHICH OCCURS DURING A READ DATA OR A READ HEADER & DATA OPERATION
                                AND IS NOT CORRECTABLE OR DOES NOT BECOME CORRECTABLE AFTER THE
                                PROGRAM HAS PERFORMED THE COMPLETE RETRY SEQUENCE ON THE BAD
                                SECTOR.

                                THE RETRY SEQUENCE IS 16 RE-READS AT TRACK CENTER AND 2 ATTEMPTS
                                BOTH AT POSITIVE AND NEGATIVE OFFSETS.


                5.2.2   SOFT ERRORS

                        A.  ECC CORRECTABLE 'DCK' ERRORS.
                        B.  'DCK' & 'ECH' ERRORS WHICH BECOME ECC CORRECTABLE DURING
                            RETRY OR WHICH ARE READ CORRECTLY DURING RETRY.
                        C.  HEADER READ ERRORS - READ DATA, READ HEADER & DATA, OR
                            WRITE DATA ORDERS.
                        D.  'DTE' ERRORS WHICH ARE CORRECTED OR WHICH BECOME ECC CORRECTABLE
                            'DCK' ERROR DURING THE RETRY SEQUENCE.



                6.      DATA CHECKING & ERROR RECOVERY
                        ------------------------------

                6.1     DATA COMPARISON

                        DATA COMPARISON OCCURS AFTER EACH 'RDDAT' (READ DATA) OR 'RDHD'
                        (READ HEADER AND DATA) OPERATION UNDER THE FOLLOWING CONDITIONS:

                        A.  THE ORDER TERMINATED WITH NO ERROR.

                        B.  THE OPERATION TERMINATED WITH 'DCK' SET AND THE ERROR
                            IS ECC CORRECTABLE OR THE SECTOR IN ERROR IS READ
                            CORRECTLY AFTER RETRY ATTEMPTS.

                6.2     VERIFICATION OF DATA WRITTEN

                        DATA VERIFICATION IS DONE EITHER THROUGH READING THE WRITTEN
                        DATA BACK AND MATCHING THE DATA WITH ONE OF THE 15 PATTERNS
                        OR THROUGH ISSUING WRITE CHECK COMMANDS.

                6.3     SECTOR REFORMATTING

                        THE PROGRAM WILL REFORMAT AN UNCORRECTABLE ERROR SECTOR IN THE
                        FOLLOWING CASES (PARAMETER 'FORMAT' MUST BE SET AND SW<00> = 0).
                        THIS PREVENTS THE SAME ERROR FROM BEING CONTINUOUSLY REPORTED.

                        A.  DATA CHECK ERRORS - EM21
                        B.  HEADER READ ERRORS - EM20, EM24, EM25, EM26, EM27
                        C.  DRIVE TIMING ERRORS - EM31
                        D.  OPERATION INCOMPLETE ERRORS - EM32
                        E.  WRITE CHECK ERRORS - EM22, EM23

                        NOTE: THE SECTOR WILL NOT BE REFORRMATTED IF THAT PARTICULAR
                              ADDRESS IS CONTAINED IN THE BAD SECTOR TABLE. (SEE SECTION 6.4)

                6.4     BAD ADDRESS FLAGGING
```

```
                    SINCE THE RM05/3/2 SUB-SYSTEM HAS AN AUTOMATIC BAD SPOT HANDLING
                    CAPABILTIY, THE PERFORMANCE EXERCISER ALLOWS THE USER TO IDENTIFY
                    UP TO 252. BAD SECTOR LOCATIONS FOR EACH DRIVE, WHEN ASSIGNING THE
                    DRIVES FOR TESTING.

                    IF ONE OF THE FOLLOWING ERRORS OCCURS AT A LOCATION IDENTIFIED BY
                    THE PROGRAM, THE PROGRAM WILL INHIBIT THE ERROR REPORT FOR THAT ERROR.

                            DATA CHECK ERRORS ('DCK')
                            WRITE CHECK ERRORS ('WCE')
                            OPERATION INCOMPLETE ERRORS ('OPI')
                            DRIVE TIMING ERRORS ('DTE')
                            HEADER READ ERRORS ('FER W/ HCRC', 'HCE W/ HCRC' OR 'HCRC')

                    WHEN A DRIVE IS ASSIGNED TO BE TESTED,THE PROGRAM READS THE
                    BAD SECTOR FILE FROM THE LAST TRACK OF THE PACK AND THEN
                    ALLOWS THE ADDITIONAL BAD SPOTS TO BE ENTERED MANUALLY.

                    THE MAXMUM NUMBER OF BAD SPOTS ALLOWED FOR EACH DRIVE, BOTH READING
                    FROM THE LAST TRACK AND ENTERING FROM KEYBOARD, IS 252..

                    THE MANUALLY ENTERED BAD SECTORS ARE NOT RECORDED TO THE BAD SECTOR
                    FILE OF THE PACK CURRENT UNDER TESTING. IF IT IS DESIRED TO RECORD
                    THE BAD SECTORS TO THE BAD SECTOR FILE, USE THE FORMAT PROGRAM
                    CZRML-. (STARTING AT LOCATION 204)


            7.      ERROR MESSAGES
                    --------------

                    DRIVE ERRORS ARE REPORTED ON THE TELETYPE OR (IF AVAILABLE) A
                    LINE PRINTER.  ALL ERROR CONDITIONS ARE REPORTED IN ERROR MESSAGES;
                    THE PROGRAM CONTAINS NO CODED ERROR HALTS.  IF THE PROGRAM HALTS
                    (ASSUMING, OF COURSE, THAT SW<15> IS NOT SET), AN UNRECOVERABLE
                    PROGRAM CONDITION HAS OCCURRED OR A CENTRAL PROCESSOR FAILURE HAS
                    OCCURRED.

                    ERROR MESSAGES ARE MADE UP OF SEVERAL LINES.  EACH TYPE OF ERROR
                    HAS SEVERAL OPTIONAL LINES WHICH MAY APPEAR WITH IT.  ALL OF THE
                    POSSIBLE ERROR MESSAGE LINES WHICH MAY APPEAR ARE GIVEN IN THE
                    SECTION DESCRIBING THE PARTICULAR ERROR HEADER.

            7.1     ERROR DESCRIPTION LINES

                    MESSAGES EM1, EM2, EM3, EM4, EM5, EM10, EM11, & EM12 ARE ALWAYS
                    DISPLAYED ON THE TTY.  THE OTHER MESSAGES ARE DISPLAYED ON EITHER
                    THE LINE PRINTER (IF AVAILABLE) OR THE TTY.

                    (THE MESSAGE TAGS ARE GIVEN FOR REFERENCE.)

                    MESSAGE
                    TAG             TEXT
                    ---             ----

                    EM1     RH11 INTERRUPT OCCURRED (RMAS=0)
```

```
                THE RH11 INTERRUPTED AND THE ATTENTION SUMMARY REGISTER
                (RMAS) WAS CLEARED.

        EM2     UNEXPECTED ATTENTION OCCURRED

                THE INDICATED DRIVE INTERRUPTED BUT THE DRIVE WAS NOT
                PERFORMING AN OPERATION.

        EM3     MASSBUS PARITY ERROR (MCPE=1)

                THE RH11 DETECTED A CONTROL BUS PARITY ERROR WHEN READING
                THE INDICATED REGISTER FROM THE INDICATED DRIVE.

        EM4     MASSBUS PARITY ERROR (PAR=1)

                THE INDICATED RM DETECTED A CONTROL BUS PARITY ERROR
                WHEN THE RH11 LOADED THE SPECIFIED REGISTER.

        EM5     ADDRESS PLUG CHANGE BIT SET

                THE 'OPE' BIT WAS SET WHEN THE INDICATED DRIVE INTERRUPTED.

        EM6     RH11 DIDN'T RESPOND TO ADDRESSING

                WHEN THE PROGRAM ADDRESSED THE RH11, NO RESPONSE WAS RECEIVED
                FROM THE INDICATED ADDRESS.

        EM10    UNCORRECTABLE MASSBUS PARITY ERROR

                THE PROGRAM HAS TRIED 3 TIMES TO READ OR WRITE THE INDICATED
                REGISTER.

        EM11    FATAL MASSBUS PARITY ERROR

                A CONTROL BUS PARITY ERROR OCCURRED WHEN THE RH11 ATTEMPTED
                TO PROCESS A PREVIOUS, DIFFERENT PARITY ERROR.

        EM12    PERSISTENT DEVICE UNSAFE

                THE DRIVE BECAME UNSAFE; DRIVE CLEAR TO THE DRIVE DID
                NOT CLEAR THE UNSAFE CONDITION.  THE PROGRAM WILL
                AUTOMATICALLY DEASSIGN THE DRIVE.  THE DRIVE CANNOT
                BE EXERCISED UNTIL THE UNSAFE CONDITION HAS BEEN
                CLEARED BY MANUAL INTERVENTION.

        EM13    OPERATION NOT COMPLETED WITHIN TIME LIMIT

                THE DRIVE DID NOT COMPLETE THE OPERATION WITHIN 1 SECOND
                AFTER THE OPERATION WAS INITIATED.

        EM14    UNIT WENT OFFLINE

                THE DRIVE WENT OFFLINE DURING THE INDICATED OPERATION.
                (THE 'MOL' BIT BECAME ZERO.)  THE PROGRAM WILL AUTOMATICALLY
                DEASSIGN THE DRIVE.  THE OPERATOR MUST REASSIGN THE DRIVE
                WITH THE 'T' COMMAND TO RE-INITIATE TESTING.
```

```
EM15    NO RESPONSE TO PORT REQUEST

        THE PROGRAM IS TESTING A DUAL PORT DRIVE WHICH HAS NOT SWITCHED
        TO THE REQUESTING PORT WITHIN 10 SECONDS AFTER PORT REQUEST
        TO THE DRIVE FROM THE REPORTING PORT.

EM20    HEADER CRC ERROR

        A HEADER CRC ERROR WAS DETECTED AT THE INDICATED DISK ADDRESS.
        THE CONTENTS OF THE HEADER ARE DISPLAYED.  THE OPERATION WILL
        BE RETRIED 3 TIMES.

EM21    DATA CHECK ('DCK') ERROR

        A DATA CHECK ERROR WAS DETECTED AT THE INDICATED SECTOR.
        THE FULL RETRY SEQUENCE (INCLUDING OFFSET) WILL BE INITIATED
        FOR THE SECTOR IN ERROR IF THE ECC HARD ERROR ('ECH) BIT
        IS SET.

EM22    WRITE CHECK ERROR - DATA CHECK ('DCK') SET

        A WRITE CHECK ERROR OCCURRED AND THE DATA CHECK ('DCK') BIT
        WAS SET.  IF 'ECH' IS NOT SET, THE OPERATION WILL BE RETRIED
        UP TO 3 TIMES; IF THE 'ECH' BIT IS SET, THE OPERATION WILL
        BE RETRIED UP TO 16 TIMES.

EM23    WRITE CHECK ERROR - DATA CHECK ('DCK') NOT SET

        A WRITE CHECK ERROR OCCURRED AND 'DCK' WAS NOT SET.  THE
        WORDS WHICH CAUSED THE ERROR ARE DISPLAYED IN THE ERROR
        MESSAGE.  THE OPERATION WILL BE RETRIED 3 TIMES.

EM24    HEADER READ ERROR - 'FMT' BIT DROPPED

        A WRITE DATA, WRITE CHECK DATA, OR A READ DATA WAS BEING
        PERFORMED AND A 'FMT' ERROR OCCURRED.  THE PROGRAM RE-READ THE
        HEADER OF THE ERROR SECTOR AND THE 'HCRC' BIT WAS SET.  THE
        CONTENTS OF THE HEADER ARE DISPLAYED.  THE OPERATION WILL
        BE RETRIED 3 TIMES.

EM25    HEADER READ ERROR - HEADER COMPARE ('HCE') ERROR

        SIMILAR TO EM24, EXCEPT THAT THE 'HCE' ERROR BIT WAS
        SET INITIALLY.  THE OPERATION WILL BE RETRIED 3 TIMES.

EM26    FORMAT ERROR ('FER')

        FORMAT ERROR OCCURRED.  WHEN THE HEADER WAS RE-READ, THE
        'HCRC' BIT WAS NOT SET.  THE CONTENTS OF THE HEADER ARE
        DISPLAYED.  THE OPERATION WILL BE RETRIED 3 TIMES.

EM27    HEADER COMPARE ('HCE') ERROR

        SIMILAR TO EM26 EXCEPT THAT THE 'HCE' BIT WAS SET INITIALLY.
        THE OPERATION WILL BE RETRIED 3 TIMES.

EM30    MISCELLANEOUS DRIVE ERROR
```

```

                THIS MESSAGE IS GIVEN FOR THE FOLLOWING ERROR BITS:
                 'AOE', 'RMR', 'ILF', OR 'ILR'

        EM31    OPERATION INCOMPLETE ('OPI') ERROR

                AN OPERATION INCOMPLETE ERROR OCCURRED AT THE INDICATED
                SECTOR.

        EM32    DRIVE TIMING ('DTE') ERROR

                DRIVE TIMING ERROR OCCURRED ON THE INDICATED SECTOR.  THE
                OPERATION WILL BE RETRIED 3 TIMES.

        EM33    PARITY ('PAR') ERROR AFTER OPERATION STARTED

                THE 'PAR' BIT WAS SET WHEN THE OPERATION WAS COMPLETED.  THE
                OPERATION WILL BE RETRIED 3 TIMES.

        EM34    WRITE CLOCK FAILURE ('WCF')

                A WRITE CLOCK FAILURE OCCURRED DURING THE OPERATION.  THE
                OPERATION WILL BE RETRIED 3 TIMES.

        EM35    INVALID ADDRESS ('IAE') ERROR

                AN INVALID ADDRESS ERROR OCCURRED DURING THE OPERATION.

        EM36    WRITE LOCK ('WLE') ERROR

                A WRITE OPERATION WAS ATTEMPTED BUT THE DRIVE WAS WRITE
                LOCKED.

        EM40    RH11 OR UNIBUS TRANSFER ERROR

                'TRE' IS SET IN THE RH11 CONTROL REGISTER AND NO DRIVE
                ERROR HAS OCCURRED.  THE OPERATION WILL BE RETRIED 3
                TIMES IF THE ERROR WAS CAUSED BY 'DLT', 'UPE', 'MXF',
                OR 'MDPE'.

        EM41    BUS ADDRESS OR WORD COUNT INCORRECT

                NO DRIVE ERROR OCCURRED BUT EITHER THE BUS ADDRESS INDICATES
                THAT AN INCORRECT NUMBER OF WORDS WERE TRANSFERED OR THE
                WORD COUNT REGISTER IS NOT ZERO.

        EM42    DATA COMPARE ERRORS - NO DRIVE ERROR DETECTED

                NO SUBSYSTEM ERROR WAS SIGNALED; HOWEVER, THE DATA DOES NOT
                COMPARE.

        EM43    CAN'T MATCH DATA READ WITH A PATTERN

                THE DATA IN THE BUFFER DOES NOT MATCH ANY OF THE STANDARD
                PATTERNS.

        EM44    ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11
```

THE OPERATION COMPLETED NORMALLY; HOWEVER, THE PROGRAM FOUND EITHER ERROR BITS IN THE RM SET OR ERROR BITS IN THE RH11 SET.

EM45 ECC LOGIC FAILURE

THE CONTENTS OF EITHER THE ECC POSITION REGISTER (RMEC1) OR THE CONTENTS OF ECC PATTERN REGISTER (RMEC2) ARE NOT VALID. THE POSITION REGISTER IS EITHER A 0 OR > 040066 OR THE PATTERN REGISTER CONTAINS ZEROS.

EM46 BUS ADDRESS OR WORD COUNT NOT CONSISTENT

THE PROGRAM WAS PROCESSING AN ERROR AND FOUND THAT THE NUMBER OF WORDS TRANSFERED AS INDICATED BY THE BUS ADDRESS REGISTER DOES NOT AGREE WITH THE TRANSFER COUNT FROM THE WORD COUNT REGISTER.

EM50 SEEK INCOMPLETE ERROR

THE DRIVE SIGNALED EITHER 'SKI' OR 'OCYL' ERROR BITS.

EM51 NOT USED

EM60 DEVICE UNSAFE

THE INDICATED DRIVE UNSAFE ERROR OCCURRED; THE ERROR WAS CLEARED BY A 'DRIVE CLEAR' INSTRUCTION.

7.2 DETAIL ERROR LINES

THE LINE NUMBERS GIVEN BELOW ARE FOR REFERENCE ONLY.

LINE 1

HH:MM:SS

'HH:MM:SS' IS THE TIME SINCE THE PROGRAM WAS STARTED.
(HOURS, MINUTES, SECONDS)

LINE 2

'PRESENT ORDER = XXXX PREVIOUS ORDER = YYYY'

MNEMONICS USED FOR THE ORDERS ARE DEFINED BELOW:

UNLOAD - UNLOAD (OCTAL 3)
SEEK - SEEK (OCTAL 5)
RECAL - RECALIBRATE (OCTAL 7)
DRVCLR - DRIVE CLEAR (OCTAL 11)
RELSE - RELEASE (OCTAL 13)
OFFSET - OFFSET (OCTAL 15)

```
                        RTC - RETURN TO CENTERLINE (OCTAL 17)
                        READIN - READIN PRESET (OCTAL 21)
                        PACK - PACK ACKNOWLEDGE (OCTAL 23)
                        SEARCH - SEARCH (OCTAL 31)
                        GETREG - GET REGISTERS (OCTAL 41)
                        SETFMT - SET FORMAT (ECI OR HCI) (OCTAL 43)
                        SELDRV - SELECT DRIVE (OCTAL 45)
                        WCKD -  WRITE CHECK DATA (OCTAL 51)
                        WCKHD - WRITE CHECK HEADER & DATA (OCTAL 53)
                        WRTDAT - WRITE DATA (OCTAL 61)
                        WRTHD - WRITE CHECK HEADER & DATA (OCTAL 63)
                        RDDAT - READ DATA (OCTAL 71)
                        RDHD -  READ HEADER & DATA (OCTAL 73)

                (DISPLAY OF THE RH/RM REGISTERS IN TWO GROUPS:
                RMCS1,RMCS2,RMDS1,RMER1,RMER2,RMER3,RMEC1, & RMEC2 FORM THE FIRST
                GROUP;  ALL THE OTHER REGISTERS ARE IN THE SECOND GROUP.
                IF SW<05> IS SET, ONLY THE REGISTERS IN THE FIRST GROUP WILL BE
                DISPLAYED.)


                THE ABOVE LINE WILL BE TYPED IF THE ERROR OCCURRED DURING
                THE NON-DATA TRANSFER PART OF THE OPERATION.

                '* ERROR AT BAD TRACK/SECTOR'

                THE ABOVE LINE WILL BE PRINTED IF A DATA ERROR OCCURES AT AN ADDRESS
                ON THE PACK WHICH THE OPERATOR HAS IDENTIFIED AS BEING BAD.  PARAMETER
                'NOTPRT' MUST BE 0 FOR THE ERROR TO BE REPORTED.

                A WORD CALLED 'STATUS' IS DISPLAYED WITH THE RM REGISTERS.  THE
                CONTENTS OF THIS WORD IDENTIFY HOW THE ERROR WAS PROCESSED BY THE
                RM DRIVE HANDLER ROUTINE.  THE BITS IN THIS WORD ARE ENCODED
                AS FOLLOWS:

                        BIT #           MEANING IF BIT IS '1'
                        -----           ---------------------

                         15             ERROR OCCURRED
                                          DONE (BIT07=0), BITS 14-9, 2, 1 SPECIFY TYPE
                                          DONE (BIT07=1), BITS 6-3 SPECIFY TYPE

                         14             DRIVE IS OFFLINE

                         12             PERSISTENT UNSAFE CONDITION EXISTS

                         11             UNCORRECTABLE PARITY ERROR OCCURRED

                         10             FATAL PARITY ERROR OCCURRED.  MASSBUS CLEAR
                                          WAS PERFORMED

                          9             OPERATION NOT COMPLETED WITHIN 1 SECOND
                                          MASSBUS CLEAR PERFORMED.  ALL OTHER
                                          OUTSTANDING OPERATIONS WERE RESTARTED.

                          7             DONE - OPERATION COMPLETED
                6               DATA ERROR OCCURRED DURING THE TRANSFER

                5               ERROR OCCURRED WHILE SEARCHING FOR THE 'TRANSFER'
                                  SECTOR OR DURING RECALIBRATE OR OFFSET COMMANDS

                4               CORRECTABLE UNSAFE CONDITION OCCURRED

                3               DRIVE ERROR OCCURRED THAT CAUSED AN AUTOMATIC
                                  RECALIBRATE SEQUENCE

                2               PORT REQUEST TIMEOUT

                1               NON-EXISTENT DRIVE REQUESTED

        LINE 3
        ------

        ERROR AT CXXX TYY SZZ  PREV ADDR = CUUU TVV SWW

        THE ACTUAL ADDRESS OF THE ERROR SECTOR AND THE PREVIOUS
        DISK ADDRESS ARE GIVEN IN THIS LINE.  CYLINDER, TRACK, &
        SECTOR ADDRESSES ARE IN DECIMAL.

        LINE 4
        -------

        PRESENT ADDR = CXXX TYY SZZ  PREV ADDR = CUUU TVV SWW

        THIS LINE IDENTIFIES THE ADDRESS WHEN THE ERROR WAS DETECTED;
        THE PREVIOUS ADDRESS IS ALSO GIVEN.  CYLINDER, TRACK, & SECTOR
        ADDRESSES ARE GIVEN IN DECIMAL.

        LINE 5
        -------

        START CYL = XXX END CYL = YYY

        THIS LINE IDENTIFIES THE STARTING CYLINDER OR A SEEK (IMPLIED)
        AND THE DESTINATION CYLINDER.  CYLINDER ADDRESSES ARE IN
        DECIMAL.

        LINE 6
        -------

        START CYL = XXX  END CYL = YYY  ACTUAL CYL = ZZZ

        THIS LINE IDENTIFIES THE STARTING CYLINDER OF AN IMPLIED SEEK,
        THE DESTINATION CYLINDER, AND THE CYLINDER THE DISK ACTUALLY
        STOPPED AT.  CYLINDER ADDRESSES ARE IN DECIMAL.

        LINE 7
        -------

        RMBA = XXXX  RMWC = YYYY

        THIS LINE GIVES THE CONTENTS OF THE RH11 BUFFER ADDRESS
        REGISTER AND THE RH11 WORD COUNT REGISTER.  THIS LINE IS
```

```
NOT PRINTED IF SW<05> IS NOT SET.

LINE 8
-------


START CYL = XXX  START TRK = YY  START SECTOR = ZZ

THIS LINE IDENTIFIES THE STARTING DISK ADDRESS OF THE PRESENT
OPERATION.  CYLINDER, TRACK, AND SECTOR VALUES ARE DECIMAL.

LINE 9
-------


RMDA = XXXX  RMCA = YYYY

THIS LINE GIVES THE CONTENTS OF THE RM TRACK AND SECTOR
ADDRESS REGISTER AND THE CONTENTS OF THE DESIRED CYLINDER
ADDRESS REGISTER.  THIS LINE IS NOT PRINTED IF SW<05> IS NOT
SET.

LINE 10
------


BUFFER ADDR = XXXX  SIZE = YYYY  ACTUAL NUMBR WRDS XFRD = ZZZZ

THIS LINE GIVES THE STARTING ADDRESS OF THE BUFFER USED FOR THE
CURRENT DATA TRANSFER OPERATION, ITS SIZE, AND THE ACTUAL NUMBER
OF WORD TRANSFERED.  THE STARTING ADDRESS OF THE BUFFER IS IN
OCTAL, THE SIZE AND WORD TRANSFERED VALUE ARE IN DECIMAL.

LINE 11
------


GOOD DATA = XXXX  BAD DATA = YYYY  SECT POS = ZZZ

THIS LINE GIVES THE GOOD DATA, THE ACTUAL DATA FROM THE DISK,
AND THE LOCATION IN THE SECTOR OF THE ACTUAL DATA.  THE SECTOR
POSITION IS IN DECIMAL.

LINE 12
-------


HEADER CONTENTS OF ERROR SECTOR = XXXX XXXX XXXX XXXX

THIS LINE GIVES THE CONTENTS OF THE HEADER OF THE SECTOR WHICH
GAVE THE ERROR.

LINE 13
-------


RMEC1 = XXXX  RMEC2 = YYYY

THIS LINE WILL BE PRINTED AFTER A SUCESSFUL RETRY OF A SECTOR
WHICH BECAME ECC CORRECTABLE DURING RETRY.

LINE 14
------
```

```

ECC CORRECTABLE WITHOUT OFFSET

THE SECTOR IN ERROR IS ECC CORRECTABLE; NO RETRY ATTEMPTS ARE
NECESSARY.

LINE 15
-------

READ CORRECTLY AT (NEG OR POS) OFFSET

THE SECTOR IN ERROR WAS READ WITHOUT ERROR AT THE INDICATED
OFFSET VALUE.

LINE 16
-------

ECC CORRECTABLE AT (NEG OR POS) OFFSET

THE SECTOR IN ERROR BECAME ECC CORRECTABLE AT THE INDICATED
OFFSET.

LINE 17
-------

CORRECTED ON X RETRY

THE OPERATION WAS PERFORMED ERROR FREE ON THE INDICATED RETRY
ATTEMPT.

LINE 18
-------

UNCORRECTABLE AFTER X RETRIES

THE OPERATION COULD NOT BE PERFORMED CORRECTLY AFTER THE
INDICATED NUMBER OF RETRY ATTEMPTS.

LINE 19
------

DIFFERENT ERROR DURING RETRY

WHILE THE PROGRAM WAS RETRYING THE ERROR, A DIFFERENT OCCURRED.
IF THIS LINE IS PRINTED, THE RH/RM REGISTERS WILL ALSO BE
PRINTED (SEE LINE 2).

LINE 20
-------

DATA COMPARISON ERRORS

A PRINTOUT OF THE DATA COMPARISON ERRORS FOLLOW THIS LINE.

LINE 21
-------

```

```
TOTAL COMPARE ERRORS = XXXX

THIS LINE GIVES THE TOTAL DATA COMPARISON ERROR COUNT.  THE
VALUE GIVEN IS IN DECIMAL.

LINE 22
-------


THE DATA COMPARED OK

THIS LINE INDICATES THE RESULTS OF THE DATA COMPARISON FOLLOWING
ECC CORRECTION.

LINE 23
--------


ECC CORRECTION RESULTS

THE PROGRAM PERFORMED ECC CORRECTION AND THE RESULTS ARE REPORTED.
THE ADDRESS IN MEMORY OF THE WORD(S) IN ERROR ARE GIVEN, THE WORD(S)
BEFORE CORRECTION AND THE WORD(S) AFTER CORRECTION ARE PRINTED.

LINE 24
-------


ERROR BURST BEGINS AT WORD XXX IN DATA FIELD OF ERROR SECTOR

THIS IS AN INFORMATIONAL LINE WHICH WILL BE PRINTED FOR 'DCK' ERRORS
WHICH ARE ECC CORRECTABLE OR WHICH BECOME ECC CORRECTABLE DURING
RETRY.  'XXX' IS THE WORD OFFSET VALUE FROM 'RMEC1' AND IS IN
DECIMAL.

LINE 25
--------


ERROR WAS NOT IN THE DATA READ -
ECC CORRECTION CAN'T BE PERFORMED

THE DATA ERROR WAS NOT IN DATA TRANSFERED TO MEMORY.

LINE 26
-------


CONTENTS OF THE ERROR SECTOR (REPORTED ABOVE)

IF SW<03> IS SET, THE SECTOR WHICH GAVE THE 'DCK', 'DTE' OR,
'WCF' ERROR OR 'HARD' DATA CHECK ERROR IS PRINTED.  THE
CONTENTS OF THE SECTOR FOLLOW THIS LINE.

LINE 27
------


ORDERS: WWW  ERRORS: X  WRDS XFR: YYYY  WRDS READ: ZZZZ

THIS IS THE LAST LINE PRINTED FOR ALL NON-POSITIONING
TYPE ERRORS.
```

'ORDERS' IS THE TOTAL NUMBER OF COMMANDS GIVEN TO THE DRIVE WHICH REPORTED THE ERROR.

'ERRORS IS THE TOTAL ERROR COUNT FOR THE DRIVE AND INCLUDES EVERY ERROR DETECTED, REGARDLESS OF TYPE.

'WRDS XFR' IS THE TOTAL NUMBER OF WORDS WRITTEN AND READ BY THE DRIVE.

'WRDS READ' IS THE TOTAL NUMBER OF WORD READ BY THE DRIVE.

LINE 28

```
ORDERS: WWWW TOTAL SEEKS: XXX TOTAL POS ERR = YYY TOTAL SKI ERR = Z
```

THIS IS THE LAST LINE PRINTED FOR ALL POSITIONING TYPE ERRORS.

'ORDERS' IS THE TOTAL NUMBER OF ORDERS GIVEN TO THE DRIVE WHICH REPORTED THE ERROR.

'TOTAL SEEKS' IS THE TOTAL NUMBER OF SEEK OPERATIONS PERFORMED BY THE DRIVE.

'TOTAL SKI ERR' IS THE TOTAL NUMBER OF 'SKI' ERRORS SIGNALED BY THE DRIVE.

8. PROGRAM DESCRIPTION

8.1 PROGRAM OPERATION

WHEN THE PROGRAM IS STARTED, PROVIDING APT TTY ENABLE BIT IS SET OR DIAGNOSTIC LOADED BY OTHER THAN APT SCRIPT MODE, ALL TABLES AND PARAMETERS ARE CLEARED OR INITIALIZED. THE PARAMETERS WHICH ARE UNDER OPERATOR TTY ENTRY CONTROL ARE CHECKED FOR VALIDITY AND CONSISTENCY. RH11 INTERRUPT ENABLE ('IE') IS SET, TTY KEYBOARD INTERRUPT ENABLE IS SET, AND THE KW11-L OR KW11-P CLOCK IS STARTED. WHEN THESE ACTIONS HAVE BEEN COMPLETED, THE PROGRAM TYPES OUT 'PROGRAM INTIALIZE COMPLETE'. COMMAND ENTRIES WILL NOW BE ACCEPTED BY THE PROGRAM

THE PROGRAM SCANS ITS INTERNAL ASSIGNMENT TABLES, LOOKING FOR:

1) DRIVES TO ASSIGN/DEASSIGN
2) PERFORMANCE SUMMARY TYPEOUT REQUESTS
3) DRIVES REQUIRING COMMAND INITIATION, BUFFER ASSIGNEMENT, OR PARAMETER SELECTION.
4) DRIVES COMPLETING CURRENT OPERATIONS.

THE PROGRAM CONTINUES SCANNING ITS TABLES UNTIL AN ENTRY IS FOUND. IN THE CASE OF THE PROGRAM AT INITIAL START, THE FIRST ENTRY WILL BE MADE BY THE OPERATOR WHEN A DRIVE IS ASSIGNED ('T' COMMAND).

WHEN A DRIVE IS ASSIGNED, THE KEYBOARD ENTRY ROUTINE VERIFIES THAT THE DRIVE IS PRESENT, IS AN RM05/3/2, AND IS ONLINE. THE ASSIGNMENT ROUTINE

THEN ISSUES A 'READIN PRESET' INSTRUCTION, SETS 'FMT16', AND ISSUES A 'RECALIBRATE' INSTRUCTION.

PARAMETERS FOR THE OPERATION ARE SELECTED AND A BUFFER IS ASSIGNED. IF THE OPERATION IS A WRITE OR WRITE CHECK ORDER, THE ASSIGNED BUFFER WILL BE FILLED WITH THE SELECTED PATTERN. (WRITE CHECK ORDERS ARE ISSUED AFTER EACH WRITE ORDER. THE WRITE CHECK ORDER USES THE PARAMETERS SELECTED FOR THE PRECEEDING WRITE ORDER.) CONTROL IS THEN PASSED TO THE COMMAND INITIATION ROUTINE.

THE COMMAND INITIATION ROUTINE FIRST LOOKS AT THE CYLINDER ADDRESS OF THE REQUESTED OPERATION. IF THE DRIVE MUST SEEK TO ANOTHER CYLINDER TO PERFORM THE OPERATION, THE PROGRAM ISSUES A SEARCH INSTRUCTION TO THE DRIVE WITH A 'TARGET' SECTOR WHICH IS 8 SECTORS EARLIER THAN THE 'TRANSFER' SECTOR. (THIS ALLOWS THE PROGRAM TO INITIATE OPERATIONS ON ANOTHER DRIVE WHILE THE PRESENT DRIVE, OR OTHER DRIVES, ARE SEARCHING FOR 'TARGET' SECTORS. ALL SEEKS ISSUED BY THE PROGRAM ARE IMPLIED SEEK SEARCH OPERATIONS.) WHEN A SEARCHING DRIVE FINDS THE 'TARGET' SECTOR AND INTERRUPTS, THE PROGRAM READS THE LOOK AHEAD REGISTER (RMLA) OF THE INTERRUPTING DRIVE AND COMPARES THE POSITION OF THE DISK WITH THAT OF THE DESIRED SECTOR.

IF OTHER DRIVES ARE WAITING ON CYLINDER, THEY ARE ALSO CHECKED. THE PROGRAM THEN ISSUES THE REQUESTED ORDER TO THE DRIVE NEAREST ITS TRANSFER SECTOR. THE DRIVES NOT SELECTED WILL HAVE ANOTHER SEARCH INITIATED. IF A DRIVE IS NOT SELECTED FOR TRANSFER AFTER THREE REVOLUTIONS OF ITS DISK, IT IS GIVEN PRIORITY OVER DRIVES WHICH HAVE NOT BEEN ON CYLINDER AS LONG.

WHEN THE DATA TRANSFER OPERATION IS COMPLETE, THE DRIVE REGISTERS ARE STORED AND A DATA TRANSFER IS INITIATED FOR A WAITING DRIVE.

IF THE OPERATION HAS BEEN COMPLETED NORMALLY, THE SAVED DRIVE REGISTERS ARE CHECKED TO VERIFY THAT NO ERROR BITS ARE SET; THE RH11 BUS ADDRESS AND WORD COUNT ADDRESS REGISTERS ARE CHECKED TO VERIFY THAT THE CORRECT NUMBER OF WORDS HAVE BEEN TRANSFERED AND THAT THE TWO REGISTERS ARE CONSISTENT WITH EACH OTHER; AND IF THE ORDER WAS A READ ORDER, THE DATA BUFFER IS COMPARED. WHEN THIS SEQUENCE IS COMPLETED, THE DRIVE IS RETURNED TO THE ASSIGNED, INACTIVE LIST. THE PROGRAM THEN INITIATES A DATA TRANSFER ON A WAITING DRIVE AND RESELECTS AND REINITIATES ANOTHER OPERATION ON THE RELEASED DRIVE.

ERRORS WHICH OCCUR ARE PROCESSED IN THE FOLLOWING ORDER. MULTIPLE ERRORS WILL BE REPORTED AS THE FIRST ERROR TYPE CHECKED.

A. ERRORS REPORTED FOR OPERATIONS WHICH HAVE NOT COMPLETED NORMALLY.

    PERSISTENT UNSAFE CONDITION - EM12
    UNCORRECTABLE MASSBUS PARITY ERROR - EM10
    FATAL MASSBUS PARITY ERROR - EM11
    OPERATION NOT COMPLETED WITHIN TIME LIMIT - EM13
    UNIT WENT OFFLINE - EM14

B. ERRORS REPORTED FOR OPERATIONS WHICH COMPLETE NORMALLY.

    CORRECTABLE UNSAFE - EM60
    DRIVE TIMING ERROR - EM32

```
                DATA CHECK ERROR - EM21
                WRITE CHECK WITH DCK SET - EM22
                HEADER CRC ERRORS - EM20
                FORMAT ERRORS - EM24, EM26
                HEADER COMPARE ERRORS - EM25, EM27
                PROGRAM DETECTED POSITIONING ERROR - EM51
                SEEK INCOMPLETE OR OFF CYLINDER ERROR - EM50
                WRITE CHECK WITHOUT 'DCK' SET - EM23
                RH11 OR UNIBUS TRANSFER ERROR - EM40
                'OPI' ERROR - EM31
                'PAR' ERROR - EM33
                'WCF' ERROR - EM34
                'IAE' ERROR - EM35
                'WLE' ERROR - EM36
                MISCELLANEOUS DRIVE ERROR - EM30

        C.      ERRORS NOT FLAGGED BY THE HARDWARE ERROR DETECTION LOGIC.

                BUS ADDRESS OR WORD COUNT INCORRECT - EM41
                DATA COMPARE ERRORS - NO DRIVE ERROR DETECTED - EM42
                CAN'T MATCH DATA READ WITH A PATTERN - EM43
                ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11 - EM44
                ECC LOGIC FAILURE - EM45
                BUS ADDRESS OR WORD COUNT NOT CONSISTENT - EM46
```

8.2 DUAL PORT OPERATION

DUAL PORT OPERATION IS NEARLY IDENTICAL TO THE OPERATION DESCRIBED IN SECTION 8.1. THE DIFFERENCES ARE IN COMMAND SEQUENCE INITIATION AND ORDER TERMINATION.

WHEN THE DUAL PORT HANDLER ROUTINE IN THE EXERCISER PROGRAM RECEIVES A REQUEST FOR A DRIVE, THE PROGRAM VERIFIES THAT THE DRIVE IS ONLINE. THE DRIVE IS SELECTED AND READ THE RMCS1 REGISTER BY TEST THE 'DVA'' BIT.
IF THE DRIVE IS IN NEUTRAL, THIS WILL SEIZE THE DRIVE. IF THE DRIVE IS SEIZED BY THE OTHER PORT, READING 'RMDS1' WILL SET 'PORT REQUEST'. THE PROGRAM CHECKS 'DVA' IN 'RMCS1'. IF THE DRIVE IS AVAILABLE AS INDICATED BY THE 'DVA' BIT, THE COMMAND SEQUENCE WILL BE INITIATED IN THE NORMAL MANNER (SEE SECTION 8.1 ABOVE).
IF 'DVA' WAS NOT SET, THE PROGRAM MAKES AN ENTRY FOR THE DRIVE IN AN INTERNAL 'PORT REQUEST PENDING' TABLE AND STARTS A 10 SECOND TIMER FOR THE DRIVE. IF THE DRIVE HAS NOT SWITCHED TO THE REQUESTING SYSTEM WITHIN THE 10 SECOND INTERVAL, THE PROGRAM REPORTS A 'NO RESPONSE TO PORT REQUEST' ERROR. NORMALLY THIS ERROR MESSAGE INDICATES A FAILURE IN THE DUAL PORT CONTROL LOGIC IN THE DRIVE BEING TESTED; HOWEVER, UNDER CERTAIN CONDITIONS (E.G. MASSBUS PARITY ERRORS BEING REPORTED ON THE OTHER SYSTEM ON A MOD33 TTY), THE OTHER PROCESSOR WAS UNABLE TO PROCESS THE DRIVE AFTER IT HAD REQUESTED THE DRIVE. THE OPERATOR MUST BE AWARE OF WHAT THE OTHER SYSTEM IS DOING AT ALL TIMES TO INTERPRET THE PORT RELATED ERROR MESSAGES PROPERLY.

AFTER A DRIVE HAS COMPLETED AN OPERATION, THE PROGRAM WILL STORE THE REGISTERS AND ISSUE A 'RELEASE' TO THE DRIVE; IF THE OPERATION TERMINATED WITH AN ERROR, THE DRIVE WILL NOT BE RELEASED UNTIL ERROR PROCESSING HAS BEEN COMPLETED.

SINGLE PORT DRIVES, DRIVES WHICH ARE IN NEUTRAL BUT NOT BEING EXERCISED BY THE OPPOSITE PORT ARE STILL TREATED AS DUAL PORT DRIVES IN THAT A RELEASE COMMAND IS ISSUED AT THE END OF NORMAL ORDER PROCESSING OR AT THE END OF ERROR PROCESSING. A RELEASE COMMAND ISSUED UNDER THESE CONDITIONS HAS NO FUNCTIONAL EFFECT ON THE OPERATION OF THE DRIVE.

8.3 SELECTION OF OPERATION VARIABLES

A. SECTOR ADDRESS SELECTION IS RANDOM BETWEEN THE VALUES IN 'MINSEC' AND 'MAXSEC'. TRACK ADDRESS SELECTION IS RANDOM BETWEEN THE VALUES IN 'MINTRK' AND 'MAXTRK'. CYLINDER ADDRESS SELECTION IS RANDOM BETWEEN 'MINCYL' AND 'MAXCYL'. IF A MINIMUM ADDRESS IS GREATER THAN THE CORRESPONDING MAXIMUM ADDRESS, THE PROGRAM WILL EXCLUDE ALL ADDRESSES BETWEEN 'MAX' AND 'MIN' FROM THE SELECTION. FOR EXAMPLE: IF 'MINTRK' IS 5 AND 'MAXTRK' IS IS 2, THEN TRACK ADDRESS SELECTION WILL EXCLUDE TRACKS 3 - 4 FROM THE SELECTION AND SELECT AN ADDRESS FROM AMONG ADDRESSES 5, 0, 1, 2.

B. THE BUFFER SIZE IS RANDOM SELECTED BETWEEN 4 - AND THE VALUE IN 'SIZE'. THE SIZE SELECTED IS WEIGHTED TO ENSURE THIS IS NECESSARY AS THE PROGRAM REQUIRES 4 LOCATIONS IN THE DATA PORTION OF THE SECTOR TO BE ABLE TO MATCH THE DATA TO A PATTERN FOR DATA COMPARISON PURPOSES.

C. THE DATA WRITTEN IS RANDOMLY SELECTED AMONG THE 15 STANDARD PATTERNS. THE PARAMETER 'PATTERN'' ENABLES THE RANDOM PATTERN SELECTION,IF THIS PARAMETER IS 0;OTHERWISE,THE DATA PATTERN INDEXED BY THE VALUE 'PATTERN'' IS SELECTED.

D. THE ORDERS ARE SELECTED RANDOMLY. WRITE CHECK DATA AND WRITE CHECK HEADER & DATA ORDERS ARE PERFORMED ONLY IF THE PREVIOUS ORDER WAS THE APPROPRIATE DATA ORDER. IF THE 'FORMAT' PARAMETER IS ZERO, THE PROGRAM WILL NOT SELECT WRITE HEADER & DATA (AND WRITE CHECK HEADER & DATA) ORDERS. WHEN THE PROGRAM SELECTS A WRITE HEADER & DATA ORDER, THE BUFFER SIZE IS FORCED TO 258.. THE PROGRAM WILL NOT PERFORM A MULTI-SECTOR FORMAT WRITE OPERATION.

E. THE FIRST ORDER PERFORMED AFTER A UNIT IS ASSIGNED WITH A 'W', OR 'R' COMMAND IS NOT RANDOMLY SELECTED. THE PARAMETERS FOR THE FIRST OPERATION ARE THE MINIMUM OR STARTING VALUES OF THE VARIABLES.

8.4 DATA PATTERNS

THE PROGRAM SELECTS ONE OF THE FOLLOWING DATA PATTERNS TO WRITE WHEN A WRITE ORDER IS SELECTED. THE ENTIRE BUFFER IS FILLED WITH THE SELECTED PATTERN. WHEN DATA IS READ FROM THE DISK, THE PROGRAM COMPARES DATA ON A SECTOR BASIS: FROM THE FIRST 4 DATA WORDS OF EACH SECTOR, THE PROGRAM MATCHES THE DATA TO ONE OF THE FOLLOWING PATTERNS.

```
PAT 1   PAT 2   PAT 3   PAT 4   PAT 5   PAT 6   PAT 7   PAT 8
```

```
        ------  ------  ------  ------  ------  ------  ------  ------
        000001  177776  000000  133331  052525  155554  026455  066666
        000003  177774  000000  133331  052525  155554  026455  066666
        000007  177770  000000  133331  052525  155554  026455  066666
        000017  177760  177777  133331  125252  155554  151322  066666
        000037  177740  177777  133331  125252  155554  151322  066666
        000077  177700  177777  133331  125252  155554  151322  066666
        000177  177600  000000  133331  052525  155554  026455  066666
        000377  177400  000000  133331  052525  155554  026455  066666
        000777  177000  177777  133331  125252  155554  151322  066666
        001777  176000  177777  133331  125252  155554  151322  066666
        003777  174000  000000  133331  052525  155554  026455  066666
        007777  170000  177777  133331  125252  155554  151322  066666
        017777  160000  000000  133331  052525  155554  026455  066666
        037777  140000  177777  133331  125252  155554  151322  066666
        077777  100000  000000  133331  052525  155554  026455  066666
        177777  000000  177777  133331  125252  155554  151322  066666

        PAT 9   PAT 10  PAT 11  PAT 12  PAT 13  PAT 14  PAT 15
        ------  ------  ------  ------  ------  ------  ------
        000001  177776  172666  077777  153333  000000  177777
        000002  177775  155555  137777  066667  177777  000000
        000004  177773  172666  157777  153333  177777  000000
        000010  177767  155555  167777  066667  177777  000000
        000020  177757  172666  173777  153333  177777  000000
        000040  177737  155555  175777  066667  177777  000000
        000100  177677  172666  176777  153333  177777  000000
        000200  177577  155555  177377  066667  177777  000000
        000400  177377  172666  177577  153333  177777  000000
        001000  176777  155555  177677  066667  177777  000000
        002000  175777  172666  177737  153333  177777  000000
        004000  173777  155555  177757  066667  177777  000000
        010000  167777  172666  177767  153333  177777  000000
        020000  157777  155555  177773  066667  177777  000000
        040000  137777  172666  177775  153333  177777  000000
        100000  077777  155555  177776  066667  177777  000000


9.1     RH/RM DRIVER
        --------------------

        THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH/RM DRIVER.

9.2     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
```

```
                        =0                      OFFLINE, DRIVE
                                                IS NOT AN RM05/3/2, OR
                                                NONEXISTENT DRIVE
                        <0                      UNSAFE

                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
                          5                     RM02
                          7                     RM05
                         -1                     NOT AN RM05/3/2

                THE 'RMINIT' ROUTINE WILL DO A READIN PRESET AND WILL SET FMT16.

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

                CALL:
                        JSR     R0,RM05                 ;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 FMT16, 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 RH/RM
                                                        ;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
```

```
9.4     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.


9.4.1   EXAMPLE - WRITE 1000. WORDS

        1$:     JSR     R0,RM05         ;CALL THE DRIVER
                WRTDPB                  ;DPB ADDRESS
                BR      1$              ;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,ERRTB5,0

9.5     RH/RM 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

        9.6     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 CHK HEADER & 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

        9.7     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.  (RH/RM
                                REGISTERS WERE NOT SAVED)

                (2) =>          REQUEST QUEUE HAS BEEN EMPTIED.  THE DRIVER
                                ISSUED A ''DRIVE CLEAR'' TO THE DRIVE.
                                NOTE: ALL RH/RM REGISTERS ARE SAVED
                                AS PER DPB+14 BEFORE THE ''DRIVE CLEAR''.

                (3) =>          REQUEST QUEUE HAS BEEN EMPTIED.  THE
                                DRIVER ISSUED A MASSBUS INIT.  ALL
                                RH/RM REGISTERS FOR THE DRIVE WERE
                                SAVED AS PER DPB+14 BEFORE THE INIT.

                (4) =>          A ''RECALIBRATE'' SHOULD BE ISSUED
```

```
                                   BEFORE ANY OTHER COMMAND.

                9.8     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)

                @
```

```
                         ;PROGRAM REVISION #001

                         .TITLE  CZRMUAO RM05/3/2 PERF EXER
                         ;*COPYRIGHT (C) 1980
                         ;*DIGITAL EQUIPMENT CORP.
                         ;*MAYNARD, MASS. 01754
                         ;*
                         ;*PROGRAM BY MIKE LEAVITT
                         ;*
                         ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                         ;*PACKAGE (MAINDEC-11-DZQAC-C4), 1980.
                         ;*
                         .SBTTL  OPERATIONAL SWITCH SETTINGS
                         ;*
                         ;*      SWITCH                 USE
                         ;*      ------           --------------------
                         ;*        15             HALT ON ERROR
                         ;*        13             INHIBIT ERROR TYPEOUTS
                         ;*        10             BELL ON ERROR
                         ;*         7             DISPLAY ALL DATA COMPARE ERRORS
                         ;*         6             DON'T CHANGE PARAMETERS (LOOP ON PRESENT VALUES)
                         ;*         5             A. PARTIAL REGISTER DISPLAY IF ERROR
                         ;*                       B. NO ECC CORRECTION RESULTS DISPLAYED IF ERROR
                         ;*         4             A. DO NOT CHECK FOR MAXIMUM ERROR COUNTS
                         ;*                       B. DO NOT DROP DRIVE AT END OF TEST
                         ;*         3             A. DISPLAY ERROR SECTOR IF 'DCK', 'DTE', OR 'WCF' ERROR
                         ;*                       B. DISPLAY SECTOR IF 'DCK' ERR UNCORRECTABLE AFTER
                         ;*                          28TH RETRY
                         ;*                       C. IF DATA COMPARE ERROR & SW07 SET, DISPLAY
                         ;*                          REMAINDER OF BUFFER
                         ;*         2             A. DO NOT TYPE UNIT STATUS AT PROGRAM START
                         ;*                       B. DO NOT TYPE PERFORMANCE REPORT AFTER SPECIFIED TIME
                         ;*         1             INHIBIT DATA COMPARSION AFTER READ ORDERS
                         ;*         0             READ ONLY MODE

                         .SBTTL  BASIC DEFINITIONS

                         ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
          001100         STACK   = 1100
          104000         ERROR   = EMT            ;;BASIC DEFINITION OF ERROR CALL
          000004         SCOPE   = IOT            ;;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
          177776         PSW=PS
          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
        001000                  SW9=SW09
        000400                  SW8=SW08
        000200                  SW7=SW07
        000100                  SW6=SW06
        000040                  SW5=SW05
        000020                  SW4=SW04
        000010                  SW3=SW03
        000004                  SW2=SW02
        000002                  SW1=SW01
        000001                  SW0=SW00

                                ;*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
        001000                  BIT9=BIT09
        000400                  BIT8=BIT08
        000200                  BIT7=BIT07
        000100                  BIT6=BIT06
        000040                  BIT5=BIT05
        000020                  BIT4=BIT04
        000010                  BIT3=BIT03
        000004                  BIT2=BIT02
        000002                  BIT1=BIT01
        000001                  BIT0=BIT00

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

                                .SBTTL  RH11/RH70 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          MDPE    = 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  RM REGISTERS

                        ;CONTROL AND STATUS 1 REGISTER. (#00)

        000001          GO      = 1                     ;GO BIT (BIT #0)
        000002          F0      = 2                     ;FUNCTION CODE BIT #1
        000004          F1      = 4                     ;FUNCTION CODE BIT #2
        000010          F2      = 10                    ;FUNCTION CODE BIT #3
        000020          F3      = 20                    ;FUNCTION CODE BIT #4
        000040          F4      = 40                    ;FUNCTION CODE BIT #5
        004000          DVA     = 4000                  ;DEVICE AVAILABLE (BIT #11)

                        ;DRIVE STATUS REGISTER (RMDS1) (#01)

        000001          OFFON   = 1                     ;OFFSET ON (BIT #0)
        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          LBT     = 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)


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

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

                                ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RMDA) (#05)


                                ;DRIVE TYPE REGISTER (RMDT) (#06)

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

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

        000100                  SC1     = 100                   ;SECTOR COUNT FIELD 0 (BIT #6)
        000200                  SC2     = 200                   ;SECTOR COUNT FIELD 1 (BIT #7)
        000400                  SC04    = 400                   ;SECTOR COUNT FIELD 2 (BIT #8)
        001000                  SC10    = 1000                  ;SECTOR COUNT FIELD 3 (BIT #9)
        002000                  SC20    = 2000                  ;SECTOR COUNT FIELD 4 (BIT #10)

                                ;SERIAL NUMBER REGISTER (RMSN) (#10)
                                ;(EACH IS CALLED BY BIT NUMBER)
                                ;OFFSET REGISTER (RMOF) (#11)

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

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

                                ;CURRENT CYLINDER ADDRESS (RMCC) (#13)
                                ;(REGISTER CURRENTLY NOT USED)

                                ;RM ERROR REGISTER #02 (RMER2) (#15)

        000010                  OPE     = 10
        000200                  DVC     = 200
        002000                  LBC     = 2000
        004000                  LSC     = 4000
        010000                  IVC     = 10000
        020000                  DPE     = 20000
        040000                  SKI     = 40000                 ;SEEK INCOMPLETE (BIT #14)

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

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

                                .SBTTL  RM DRIVER COMMANDS

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

        176700                  ABASE   = 176700
        000254                  AVECT1  = 254



                                .SBTTL  TRAP CATCHER

        000000                          .=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
        000174                          .=174
000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER

                                .SBTTL  STARTING ADDRESS(ES)
```

```
 000200  000137  003636             JMP     @#START1                   ;;JUMP TO STARTING ADDRESS OF PROGRAM

 000204  000137  003626             JMP     @#START             ;CHANGE THE RH ADDRESS


                             .SBTTL  ACT11 HOOKS

                             ;;*************************************************************
                             ;HOOKS REQUIRED BY ACT11
         000210                     $SVPC=.                     ;SAVE PC
         000046                     .=46
 000046  030536                     $ENDAD                      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
         000052                     .=52
 000052  040000                     .WORD   40000               ;;2)SET LOC.52 TO 40000
         000210                     .=$SVPC                     ;; RESTORE PC

         001100              .=1100
                             .SBTTL  APT PARAMETER BLOCK

                             ;;*************************************************************
                             ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                             ;;*************************************************************
         001100                     .$X=.    ;;SAVE CURRENT LOCATION
         000024                     .=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.
 000044  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.

 001100                      $APTHD:
 001100  000000              $HIBTS: .WORD   0        ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 001102  001206              $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
 001104  000264              $TSTM:  .WORD   180.     ;;RUN TIM OF LONGEST TEST
 001106  000264              $PASTM: .WORD   180.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
 001110  000264              $UNITM: .WORD   180.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
 001112  000032                      .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
         001114              TAB.XY=.         ;CMTAGSTARING ADDRESS
```

```
     0                                         .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                         $TMP0:  .WORD   0                       ;;USER DEFINED
        001176     207      377      377       $BELL:  .ASCIZ  <207><377><377>         ;;CODE FOR BELL
        001202     077                         $QUES:  .ASCII  /?/                     ;;QUESTION MARK
        001203     015                         $CRLF:  .ASCII  <15>                    ;;CARRIAGE RETURN
        001204     012      000                $LF:    .ASCIZ  <12>                    ;;LINE FEED
                                               ;;*************************************************************
                                               .SBTTL  APT MAILBOX-ETABLE

                                               ;;*************************************************************
                                               .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
        001216  000000                         $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
        001220  000000                         $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
        001222  000000                         $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
        001224  000000                         $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
        001226                                 $ETABLE:                ;;APT ENVIRONMENT TABLE
```

```
001226     000                   $ENV:    .BYTE    AENV      ;;ENVIRONMENT BYTE
001227     000                   $ENVM:   .BYTE    AENVM     ;;ENVIRONMENT MODE BITS
001230  000000                   $SWREG:  .WORD    ASWREG    ;;APT SWITCH REGISTER
001232  000000                   $USWR:   .WORD    AUSWR     ;;USER SWITCHES
001234  000000                   $CPUOP:  .WORD    ACPUOP    ;;CPU TYPE,OPTIONS
                                 ;*                          BITS 15-11=CPU TYPE
                                 ;*                                  11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                 ;*                                  11/70=06,PDQ=07,Q=10
                                 ;*                          BIT 10=REAL TIME CLOCK
                                 ;*                          BIT  9=FLOATING POINT PROCESSOR
                                 ;*                          BIT  8=MEMORY MANAGEMENT
001236     000                   $MAMS1:  .BYTE    AMAMS1    ;;HIGH ADDRESS,M.S. BYTE
001237     000                   $MTYP1:  .BYTE    AMTYP1    ;;MEM. TYPE,BLK#1
                                 ;*                          MEM.TYPE BYTE   --  (HIGH BYTE)
                                 ;*                                  900 NSEC CORE=001
                                 ;*                                  300 NSEC BIPOLAR=002
                                 ;*                                  500 NSEC MOS=003
001240  000000                   $MADR1:  .WORD    AMADR1    ;;HIGH ADDRESS,BLK#1
                                 ;*                            MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
001242     000                   $MAMS2:  .BYTE    AMAMS2    ;;HIGH ADDRESS,M.S. BYTE
001243     000                   $MTYP2:  .BYTE    AMTYP2    ;;MEM.TYPE,BLK#2
001244  000000                   $MADR2:  .WORD    AMADR2    ;;MEM.LAST ADDRESS,BLK#2
001246     000                   $MAMS3:  .BYTE    AMAMS3    ;;HIGH ADDRESS,M.S.BYTE
001247     000                   $MTYP3:  .BYTE    AMTYP3    ;;MEM.TYPE,BLK#3
001250  000000                   $MADR3:  .WORD    AMADR3    ;;MEM.LAST ADDRESS,BLK#3
001252     000                   $MAMS4:  .BYTE    AMAMS4    ;;HIGH ADDRESS,M.S.BYTE
001253     000                   $MTYP4:  .BYTE    AMTYP4    ;;MEM.TYPE,BLK#4
001254  000000                   $MADR4:  .WORD    AMADR4    ;;MEM.LAST ADDRESS,BLK#4
001256  000254                   $VECT1:  .WORD    AVECT1    ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001260  000000                   $VECT2:  .WORD    AVECT2    ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001262  176700                   $BASE:   .WORD    ABASE     ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001264  000000                   $DEVM:   .WORD    ADEVM     ;;DEVICE MAP
001266  000000                   $CDW1:   .WORD    ACDW1     ;;CONTROLLER DESCRIPTION WORD#1
001270  000000                   $CDW2:   .WORD    ACDW2     ;;CONTROLLER DESCRIPTION WORD#2
001272                           $ETEND:
                                 .MEXIT
```

```
     0                                          .SBTTL  USER DEFINED TAGS

        001272  176700                          $RMADR: .WORD   176700  ;FIRST ADDRESS OF RH/RM REGISTERS
        001274  000254                          $RMVEC: .WORD   254     ;VECTOR ADDRESS
        001276  172540                          $LKCSR: .WORD   172540  ;ADDR OF KW11-P STATUS REGISTER
        001300  172542                          $LKCSB: .WORD   172542  ;ADDR OF KW11-P COUNTER BUFFER
        001302  000104                          $LPVEC: .WORD   104     ;ADDR OF KW11-P VECTOR
        001304  177546                          $LKS:   .WORD   177546  ;ADDR OF KW11-L STATUS REGISTER
        001306  000100                          $LLVEC: .WORD   100     ;ADDR OF KW11-L VECTOR
        001310  177777                          PCLOCK: .WORD   -1      ;'0' IF KW11-P IS ON SYSTEM
        001312  177777                          CLKFLG: .WORD   -1      ;'0' IF A CLOCK IS AVAILABLE
        001314  000074                          HZ:     .WORD   60.     ;74(8) IF 60 HZ SYSTEM, 62(8) IF 50 HZ SYSTEM
        001316  000000                          STATIN: .WORD   0       ;'TYPE STATISTICS' INDICATOR
        001320  000000                          PACK:   .WORD   0       ;'W ' COMMAND INDICATOR
        001322  000000  000000  000000          DATE:   .WORD   0,0,0,0,0       ;OPERATOR ENTERED DATE
        001334  000000  000000  000000          OPERID: .WORD   0,0,0,0 ;OPERATOR ID

                001220                          DRIVE   =$UNIT          ;DRIVE # STORAGE: ERRORS 1-5 & 10
                                                                        ;SAME AS USED IN APT
        001344  000000                          ATTN:   .WORD   0       ;ATTN REG STORAGE: ERRORS 1-5 & 10
        001346  000000                          UNIT:   .WORD   0       ;DRIVE # STORAGE FOR PRINTOUT
        001350  000000                          MASK:   .WORD   0       ;ERROR RETRY REGISTER MASK
        001352     000     000                  RETRY:  .BYTE   0,0     ;ERROR RETRY LIMIT IN THE LOWER BYTE
                                                                        ;RETRY COUNT IN THE UPPER BYTE
        001354  000003                          FAIRNS: .WORD   3       ;MAXIMUM TIME IN QUEUE VALUE
        001356  000000                          LSTAD:  .WORD   0       ;STORE LAST MEMORY ADDRESS HERE
        001360  000000                          CHGADR: .WORD   0       ;CHANGE RH/RM UNIBUS ADDRESS FLAG
        001362  000000                          CFLAG:  .WORD   0       ;'CONTROL C' FLAG
        001364  000000                          BADSEC: .WORD   0       ;BAD SECTOR/TRACK FLAG
        001366  000000                          HOUR:   .WORD   0       ;HOUR COUNT STORED HERE (MAXIMUM - 999.)
        001370  000000                          MINUTE: .WORD   0       ;MINUTE'S COUNT STORED HERE
        001372  000000                          SECOND: .WORD   0       ;SECOND'S COUNT STORED HERE
        001374  000000                          SIXTEE: .WORD   0       ;TIMER ROUTINE COUNTER (FOR ONE SECOND)
        001376  177777                          ZROIND: .WORD   -1      ;ZERO INDICATOR FOR THE DATA COMPARE ROUTINE
        001400     000                          FRSTER: .BYTE   0       ;DATA COMPARE ERROR FLAG
                                                                        ;IF > 0, PROCESSING 'DCKER' OR CAN'T MATCH PATTERN
                                                                        ;IF < 0, MISCOMPARSION FOUND
        001401     000                                  .BYTE   0       ;MISCOMPARSION OR CAN'T MATCH PATTERN FLAG
                                                                        ;IF < 0, ERROR IN BUFFER
        001402  000000                          SAVER1: .WORD   0       ;SAVE R1 HERE
        001404  000000                          SAVER5: .WORD   0       ;SAVE R5 HERE
        001406  000000                          ERCTR:  .WORD   0       ;NUMBER OF ERRORS
        001410  000000                          LIMIT:  .WORD   0       ;DISPLAY LIMIT
        001412  000000                          CMCNT:  .WORD   0       ;WORD COUNT
        001414  000000                          CMCYL:  .WORD   0       ;CYLINDER ADDRESS
        001416     000                          CMSEC:  .BYTE   0       ;SECTOR ADDRESS
        001417     000                          CMTRK:  .BYTE   0       ;TRACK ADDRESS
        001420  000000                          ECBIT:  .WORD   0       ;ERROR BURST BIT OFFSET
        001422  000000                          ECSEC:  .WORD   0       ;ERROR BURST WORD OFFSET (RELATIVE TO SECTOR)
        001424  000000                          ECMSK0: .WORD   0       ;CORRECTION MASK FOR FIRST ERROR WORD
        001426  000000                          ECMSK1: .WORD   0       ;CORRECTION MASK FOR SECOND ERROR WORD
        001430  000000                          ECWRD:  .WORD   0       ;LOCATION OF FIRST ERROR WORD
        001432  000000                          ECGD:   .WORD   0       ;GOOD DATA, FIRST WORD
        001434  000000                          ECBAD0: .WORD   0       ;BAD DATA, FIRST WORD
        001436  000000                          ECWRD1: .WORD   0       ;LOCATION OF SECOND ERROR WORD
        001440  000000                          ECGD1:  .WORD   0       ;GOOD DATA, SECOND WORD
        001442  000000                          ECBAD1: .WORD   0       ;BAD DATA, SECOND WORD
     001444  001465                     CYLIMT: .WORD   821.     ;CYLINDER ADDRESS LIMIT
     001446  000037                     SECLMT: .WORD   31.      ;SECTOR ADDRESS LIMIT
     001450  000004                     TRKLMT: .WORD   4.       ;TRACK ADDRESS LIMIT, RM02/3 = 4. AND RM05 = 18.
     001452  000000                     XXDP:   .WORD   0        ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
                                                                 ;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
                                                                 ;'XXDP' DEVICE CODE FOR THE RM05/3/2.

                                        .SBTTL  COMMON PARAMETERS

     001454  002740                     ENDCON: .WORD   002740   ;1.875 X 10^8 WORDS (10) (3 X 10^9 BITS)
     001456  005455                             .WORD   005455   ;MSW
     001460  143300                     ENDSEK: .WORD   143300   ;3 X 10^6 SEEKS  (LSW)
     001462  000055                             .WORD   55       ;MSW
     001464  000001                     PASCNT: .WORD   1        ;NUMBER OF PASSES TO END OF TEST
     001466  000000                     MAXDL:  .WORD   0        ;MAXIMUM DATA TRANFER SIZE IN WORDS
                                                                 ;(FILLED BY PROGRAM AT STARTUP OR BY OPERATOR
                                                                 ;DURING PARAMETER ENTRY DIALOG.)
     001470  000144                     MAXER:  .WORD   100.     ;MAXIMUM ERRORS - 100(10)
     001472  000170  000000             INTRVL: .WORD   120.,0   ;FIRST WORD IS THE PERFORMANCE TYPEOUT INTERVAL
                                                                 ;(IN MINUTES). SECOND WORD IS THE INTERVAL COUNTER.
     001476  000004                     CMPLMT: .WORD   4        ;NUMBER OF COMPARE ERRORS TYPED OUT
     001500  000001                     FORMAT: .WORD   1        ;IF NOT EQ 0, ALLOW WRITE HEADER & DATA ORDERS
                                                                 ;IF EQ 0, DO NOT ALLOW WRITE HEADER & DATA ORDERS
     001502  000000                     WCSEL:  .WORD   0        ;IF EQ TO 0, GENERATE A RANDOM WORD COUNT
                                                                 ;  FOR THE OPERATION.
                                                                 ;IF NOT EQ TO 0, USE THE VALUE IN 'MAXDL' FOR
                                                                 ;  THE WORD COUNT
     001504  000003                     RATIO:  .WORD   3        ;READ/WRITE RATIO [RANGE 0 - 7]
                                                                 ;0 - 0/8        (READ/WRITE)
                                                                 ;1 - 7/1
                                                                 ;2 - 6/2
                                                                 ;3 - 5/3
                                                                 ;4 - 4/4
                                                                 ;5 - 3/5
                                                                 ;6 - 2/6
                                                                 ;7 - 1/7
     001506  000001                     AUTOCK: .WORD   1        ;IF NOT EQ 0, DO AN APPROPRITE WRITE
                                                                 ;CHECK AFTER EACH WRITE ORDER.
                                                                 ;IF EQ 0, SELECT WRITE CHECK ORDERS
                                                                 ;RANDOMLY.
     001510  000001                     NOTPRT: .WORD   1        ;IF EQ 1, DO NOT PRINT DATA ERROR MESSAGES
                                                                 ;ASSOCIATED WITH OPERATOR SPECIFIED
                                                                 ;BAD PACK AREAS.
                                                                 ;IF NOT EQ 0, PRINT ERROR MESSAGES RELATING TO
                                                                 ;THESE AREAS.
     001512  000001                     ENDET:  .WORD   1        ;IF NOT EQ 0, END OF PASS DETERMINED
                                                                 ;BY THE 'WORDS READ' COUNT.
                                                                 ;IF EQ 0, END OF PASS DETERMINED
                                                                 ;BY THE SEEK COUNT.
     001514  000000                     PATTEN: .WORD   0        ;IF EQ 0,RANDOMLY SELECT DATA PATTERN
                                                                 ;IF NOT EQ 0,SELECT ONE SET OF PATTERN
                                                                 ;POINTED BY THE 'PATTEN'.
     001516  000000                     HEADER: .WORD   0        ;IF EQ TO 0,RANDOMLY SELECT DATA BLOCK
                                                                 ;ADDRESS. IF NOT EQU 0,SEQUENTIALLY
                                                                 ;SELECT DATA BLOCK ADDRESS

                                        .SBTTL  VALUES FOR FIRST OPERATION
```

```
001520  000010                  BEGPAT: .WORD   10      ;STARTING PATTERN CODE [RANGE 1 - 17 (OCTAL)]
001522  000004                  BEGCOD: .WORD   4       ;STARTING COMMAND CODE [RANGE 0 - 5]
                                                        ;0 = WRITE CHECK DATA ('WCKD')
                                                        ;1 = WRITE CHECK HEADER & DATA ('WCHKHD')
                                                        ;2 = WRITE DATA ('WRTDAT')
                                                        ;3 = WRITE HEADER & DATA ('WRTHD')
                                                        ;4 = READ DATA ('RDDAT')
                                                        ;5 = READ HEADER & DATA ('RDHD')
001524  000400                  BEGSIZ: .WORD   400     ;STARTING RECORD SIZE [RANGE 4 - MAXMEM]

                                .SBTTL  TABLES, CONSTANTS, AND VARIABLE LOCATIONS

                                ;LIST OF DRIVES PERFORMING COMMANDS
001526  000000                  ORDERQ: .WORD   0
001530  000000                          .WORD   0
001532  000000                          .WORD   0
001534  000000                          .WORD   0
001536  000000                          .WORD   0
001540  000000                          .WORD   0
001542  000000                          .WORD   0
001544  000000                          .WORD   0
001546  000000                          .WORD   0

001550  000000                  ASNLST: .WORD   0                       ;A BIT SET IS AN ASSIGNED DRIVE

                                ;ADDRESSES OF DRIVES TO BE DEASSIGNED
001552  000000                  DUNIT:  .WORD   0
001554  000000                          .WORD   0
001556  000000                          .WORD   0
001560  000000                          .WORD   0
001562  000000                          .WORD   0
001564  000000                          .WORD   0
001566  000000                          .WORD   0
001570  000000                          .WORD   0
001572  000000                          .WORD   0

                                ;ADDRESSES OF NEWLY ASSIGNED DRIVES
001574  000000                  NEWUNT: .WORD   0
001576  000000                          .WORD   0
001600  000000                          .WORD   0
001602  000000                          .WORD   0
001604  000000                          .WORD   0
001606  000000                          .WORD   0
001610  000000                          .WORD   0
001612  000000                          .WORD   0
001614  000000                          .WORD   0

                                ;LIST OF DRIVES WAITING FOR BUFFERS/PARAMETERS
001616  000000                  AVAIL:  .WORD   0
001620  000000                          .WORD   0
001622  000000                          .WORD   0
001624  000000                          .WORD   0
001626  000000                          .WORD   0
001630  000000                          .WORD   0
001632  000000                          .WORD   0
001634  000000                          .WORD   0
```

```
     001636  000000                            .WORD   0

                                        ;LIST OF DRIVES WAITING FOR BUFFERS
     001640  000000                     WAIT:  .WORD   0
     001642  000000                            .WORD   0
     001644  000000                            .WORD   0
     001646  000000                            .WORD   0
     001650  000000                            .WORD   0
     001652  000000                            .WORD   0
     001654  000000                            .WORD   0
     001656  000000                            .WORD   0
     001660  000000                            .WORD   0

                                        ;LIST OF DRIVES WAITING FOR NEXT PARAMETERS
     001662  000000                     PARQ:  .WORD   0
     001664  000000                            .WORD   0
     001666  000000                            .WORD   0
     001670  000000                            .WORD   0
     001672  000000                            .WORD   0
     001674  000000                            .WORD   0
     001676  000000                            .WORD   0
     001700  000000                            .WORD   0
     001702  000000                            .WORD   0

                                        ;BUFFER ALLOCATION TABLE ENTRY COUNT
     001704  000000                     BUFTBL: .WORD  0
     001706  000000  000000                    .WORD   0,0
     001712  000000  000000                    .WORD   0,0
     001716  000000  000000                    .WORD   0,0
     001722  000000  000000                    .WORD   0,0
     001726  000000  000000                    .WORD   0,0
     001732  000000  000000                    .WORD   0,0
     001736  000000  000000                    .WORD   0,0
     001742  000000  000000                    .WORD   0,0
     001746  000000  000000                    .WORD   0,0
     001752  000000  000000                    .WORD   0,0
     001756  000000  000000                    .WORD   0,0
     001762  000000  000000                    .WORD   0,0
     001766  000000  000000                    .WORD   0,0
     001772  000000  000000                    .WORD   0,0
     001776  000000  000000                    .WORD   0,0
     002002  000000  000000                    .WORD   0,0
     002006  000000  000000                    .WORD   0,0
     002012  000000  000000                    .WORD   0,0
     002016  000000  000000                    .WORD   0,0
     002022  000000  000000                    .WORD   0,0
     002026  000000  000000                    .WORD   0,0
     002032  000000  000000                    .WORD   0,0
     002036  000000  000000                    .WORD   0,0
     002042  000000  000000                    .WORD   0,0
     002046  000000  000000                    .WORD   0,0
     002052  000000  000000                    .WORD   0,0
     002056  000000  000000                    .WORD   0,0
     002062  000000  000000                    .WORD   0,0
     002066  000000  000000                    .WORD   0,0
     002072  000000  000000                    .WORD   0,0
     002076  000000  000000                    .WORD   0,0
```

```
002102  000000  000000                      .WORD   0,0

002106  044210                   BLKADR: .WORD   DRIVE0  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 0
002110  046376                           .WORD   DRIVE1  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 1
002112  050564                           .WORD   DRIVE2  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 2
002114  052752                           .WORD   DRIVE3  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 3
002116  055140                           .WORD   DRIVE4  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 4
002120  057326                           .WORD   DRIVE5  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 5
002122  061514                           .WORD   DRIVE6  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 6
002124  063702                           .WORD   DRIVE7  ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 7

002126     151                   COMTBL: .BYTE   WCKD    ;WRITE CHECK DATA
002127     153                           .BYTE   WCKHD   ;WRITE CHECK HEADER AND DATA
002130     161                           .BYTE   WRTDAT  ;WRITE DATA
002131     163                           .BYTE   WRTHD   ;WRITE HEADER AND DATA
002132     171                           .BYTE   RDDAT   ;READ DATA
002133     173                           .BYTE   RDHD    ;READ HEADER AND DATA

002134     002                   OPTBL:  .BYTE   2       ;UNLOAD
002135     004                           .BYTE   4       ;SEEK
002136     006                           .BYTE   6       ;RECAL
002137     010                           .BYTE   10      ;DRIVE CLEAR
002140     012                           .BYTE   12      ;RELEASE
002141     014                           .BYTE   14      ;OFFSET
002142     016                           .BYTE   16      ;RETURN TO CENTERLINE
002143     020                           .BYTE   20      ;READIN PRESET
002144     022                           .BYTE   22      ;PACK ACKNOWLEDGE
002145     030                           .BYTE   30      ;SEARCH
002146     050                           .BYTE   50      ;WRITE CHECK DATA
002147     052                           .BYTE   52      ;WRITE CHECK HEADER AND DATA
002150     060                           .BYTE   60      ;WRITE DATA
002151     062                           .BYTE   62      ;WRITE HEADER AND DATA
002152     070                           .BYTE   70      ;READ DATA
002153     072                           .BYTE   72      ;READ HEADER AND DATA
002154     377                           .BYTE   -1      ;TERMINATOR

                                 .EVEN

002156     125     116     114   MNTBL:  .ASCIZ  /UNLOAD /
002166     123     105     105           .ASCIZ  /SEEK   /
002176     122     105     103           .ASCIZ  /RECAL  /
002206     104     122     126           .ASCIZ  /DRVCLR /
002216     122     105     114           .ASCIZ  /RELSE  /
002226     117     106     106           .ASCIZ  /OFFSET /
002236     122     124     103           .ASCIZ  /RTC    /
002246     122     105     101           .ASCIZ  /READIN /
002256     120     101     103           .ASCIZ  /PACK   /
002266     123     105     101           .ASCIZ  /SEARCH /
002276     127     103     113           .ASCIZ  /WCKD   /
002306     127     103     113           .ASCIZ  /WCKHD  /
002316     127     122     124           .ASCIZ  /WRTDAT /
002326     127     122     124           .ASCIZ  /WRTHD  /
002336     122     104     104           .ASCIZ  /RDDAT  /
002346     122     104     110           .ASCIZ  /RDHD   /
002356     116     117     116           .ASCIZ  /NONE   /
```

```
    002366       101       106       124   OFMSG0: .ASCIZ  /AFTER RETRY WITHOUT OFFSET/
    002421       101       124       040   OFMSG1: .ASCIZ  /AT NEGATIVE OFFSET/
    002444       101       124       040   OFMSG2: .ASCIZ  /AT POSITIVE OFFSET/

                                           .EVEN

    002470       000                       OFFCOD: .BYTE   0        ;OFFSET CODE TABLE
    002471       001                               .BYTE   1        ;NUMBER FOR NEGATIVE OFFSET (DIR = OUT)
    002472       000                               .BYTE   0        ;NUMBER FOR POSITIVE OFFSET (DIR = IN)

                                           .EVEN

    002474  002366                         OFMTBL: .WORD   OFMSG0   ;1ST OFFSET MESSAGE
    002476  002421                                 .WORD   OFMSG1   ;2ND OFFSET MESSAGE
    002500  002444                                 .WORD   OFMSG2   ;3RD OFFSET MESSAGE

                                           .SBTTL  DATA PATTERNS

                                           ;STANDARD DATA PATTERN POINTER TABLE
    002502  002546                         STNDAT: .WORD   DATA0    ;STANDARD DATA PATTERN 0
    002504  002606                                 .WORD   DATA1    ;STANDARD DATA PATTERN 1
    002506  002646                                 .WORD   DATA2    ;STANDARD DATA PATTERN 2
    002510  002706                                 .WORD   DATA3    ;STANDARD DATA PATTERN 3
    002512  002746                                 .WORD   DATA4    ;STANDARD DATA PATTERN 4
    002514  003006                                 .WORD   DATA5    ;STANDARD DATA PATTERN 5
    002516  003046                                 .WORD   DATA6    ;STANDARD DATA PATTERN 6
    002520  003106                                 .WORD   DATA7    ;STANDARD DATA PATTERN 7
    002522  003146                                 .WORD   DATA8    ;STANDARD DATA PATTERN 8
    002524  003206                                 .WORD   DATA9    ;STANDARD DATA PATTERN 9
    002526  003246                                 .WORD   DATA10   ;STANDARD DATA PATTERN 10
    002530  003306                                 .WORD   DATA11   ;STANDARD DATA PATTERN 11
    002532  003346                                 .WORD   DATA12   ;STANDARD DATA PATTERN 12
    002534  003406                                 .WORD   DATA13   ;STANDARD DATA PATTERN 13
    002536  003446                                 .WORD   DATA14   ;STANDARD DATA PATTERN 14
    002540  003506                                 .WORD   DATA15   ;STANDARD DATA PATTERN 15
    002542  002546                                 .WORD   ZEROS    ;ALL 0'S PATTERN
    002544  003450                                 .WORD   ONES     ;ALL 1'S PATTERN

    002546                                 ZEROS:
    002546  000000                         DATA0:  .WORD   0        ;ALL 0'S DATA PATTERN
    002550  000000                                 .WORD   0
    002552  000000                                 .WORD   0
    002554  000000                                 .WORD   0
    002556  000000                                 .WORD   0
    002560  000000                                 .WORD   0
    002562  000000                                 .WORD   0
    002564  000000                                 .WORD   0
    002566  000000                                 .WORD   0
    002570  000000                                 .WORD   0
    002572  000000                                 .WORD   0
    002574  000000                                 .WORD   0
    002576  000000                                 .WORD   0
    002600  000000                                 .WORD   0
    002602  000000                                 .WORD   0
    002604  000000                                 .WORD   0

    002606  000001                         DATA1:  .WORD   000001   ;STANDARD PATTERN 1
```

```
      002610  000003                  .WORD   000003
      002612  000007                  .WORD   000007
      002614  000017                  .WORD   000017
      002616  000037                  .WORD   000037
      002620  000077                  .WORD   000077
      002622  000177                  .WORD   000177
      002624  000377                  .WORD   000377
      002626  000777                  .WORD   000777
      002630  001777                  .WORD   001777
      002632  003777                  .WORD   003777
      002634  007777                  .WORD   007777
      002636  017777                  .WORD   017777
      002640  037777                  .WORD   037777
      002642  077777                  .WORD   077777
      002644  177777                  .WORD   177777

      002646  177776          DATA2:  .WORD   177776  ;STANDARD PATTERN 2
      002650  177774                  .WORD   177774
      002652  177770                  .WORD   177770
      002654  177760                  .WORD   177760
      002656  177740                  .WORD   177740
      002660  177700                  .WORD   177700
      002662  177600                  .WORD   177600
      002664  177400                  .WORD   177400
      002666  177000                  .WORD   177000
      002670  176000                  .WORD   176000
      002672  174000                  .WORD   174000
      002674  170000                  .WORD   170000
      002676  160000                  .WORD   160000
      002700  140000                  .WORD   140000
      002702  100000                  .WORD   100000
      002704  000000                  .WORD   000000

      002706  000000          DATA3:  .WORD   000000  ;STANDARD PATTERN 3
      002710  000000                  .WORD   000000
      002712  000000                  .WORD   000000
      002714  177777                  .WORD   177777
      002716  177777                  .WORD   177777
      002720  177777                  .WORD   177777
      002722  000000                  .WORD   000000
      002724  000000                  .WORD   000000
      002726  177777                  .WORD   177777
      002730  177777                  .WORD   177777
      002732  000000                  .WORD   000000
      002734  177777                  .WORD   177777
      002736  000000                  .WORD   000000
      002740  177777                  .WORD   177777
      002742  000000                  .WORD   000000
      002744  177777                  .WORD   177777

      002746  133331          DATA4:  .WORD   133331  ;STANDARD PATTERN 4
      002750  133331                  .WORD   133331
      002752  133331                  .WORD   133331
      002754  133331                  .WORD   133331
      002756  133331                  .WORD   133331
      002760  133331                  .WORD   133331
      002762  133331                  .WORD   133331
```

```
        002764  133331                  .WORD   133331
        002766  133331                  .WORD   133331
        002770  133331                  .WORD   133331
        002772  133331                  .WORD   133331
        002774  133331                  .WORD   133331
        002776  133331                  .WORD   133331
        003000  133331                  .WORD   133331
        003002  133331                  .WORD   133331
        003004  133331                  .WORD   133331

        003006  052525          DATA5:  .WORD   052525  ;STANDARD PATTERN 5
        003010  052525                  .WORD   052525
        003012  052525                  .WORD   052525
        003014  125252                  .WORD   125252
        003016  125252                  .WORD   125252
        003020  125252                  .WORD   125252
        003022  052525                  .WORD   052525
        003024  052525                  .WORD   052525
        003026  125252                  .WORD   125252
        003030  125252                  .WORD   125252
        003032  052525                  .WORD   052525
        003034  125252                  .WORD   125252
        003036  052525                  .WORD   052525
        003040  125252                  .WORD   125252
        003042  052525                  .WORD   052525
        003044  125252                  .WORD   125252

        003046  155554          DATA6:  .WORD   155554  ;STANDARD PATTERN 6
        003050  155554                  .WORD   155554
        003052  155554                  .WORD   155554
        003054  155554                  .WORD   155554
        003056  155554                  .WORD   155554
        003060  155554                  .WORD   155554
        003062  155554                  .WORD   155554
        003064  155554                  .WORD   155554
        003066  155554                  .WORD   155554
        003070  155554                  .WORD   155554
        003072  155554                  .WORD   155554
        003074  155554                  .WORD   155554
        003076  155554                  .WORD   155554
        003100  155554                  .WORD   155554
        003102  155554                  .WORD   155554
        003104  155554                  .WORD   155554

        003106  026455          DATA7:  .WORD   026455  ;STANDARD PATTERN 7
        003110  026455                  .WORD   026455
        003112  026455                  .WORD   026455
        003114  151322                  .WORD   151322
        003116  151322                  .WORD   151322
        003120  151322                  .WORD   151322
        003122  026455                  .WORD   026455
        003124  026455                  .WORD   026455
        003126  151322                  .WORD   151322
        003130  151322                  .WORD   151322
        003132  026455                  .WORD   026455
        003134  151322                  .WORD   151322
        003136  026455                  .WORD   026455
```

```
      003140  151322                         .WORD   151322
      003142  026455                         .WORD   026455
      003144  151322                         .WORD   151322

      003146  066666                DATA8:   .WORD   066666  ;STANDARD PATTERN 8
      003150  066666                         .WORD   066666
      003152  066666                         .WORD   066666
      003154  066666                         .WORD   066666
      003156  066666                         .WORD   066666
      003160  066666                         .WORD   066666
      003162  066666                         .WORD   066666
      003164  066666                         .WORD   066666
      003166  066666                         .WORD   066666
      003170  066666                         .WORD   066666
      003172  066666                         .WORD   066666
      003174  066666                         .WORD   066666
      003176  066666                         .WORD   066666
      003200  066666                         .WORD   066666
      003202  066666                         .WORD   066666
      003204  066666                         .WORD   066666

      003206  000001                DATA9:   .WORD   000001  ;STANDARD PATTERN 9
      003210  000002                         .WORD   000002
      003212  000004                         .WORD   000004
      003214  000010                         .WORD   000010
      003216  000020                         .WORD   000020
      003220  000040                         .WORD   000040
      003222  000100                         .WORD   000100
      003224  000200                         .WORD   000200
      003226  000400                         .WORD   000400
      003230  001000                         .WORD   001000
      003232  002000                         .WORD   002000
      003234  004000                         .WORD   004000
      003236  010000                         .WORD   010000
      003240  020000                         .WORD   020000
      003242  040000                         .WORD   040000
      003244  100000                         .WORD   100000

      003246  177776                DATA10:  .WORD   177776  ;STANDARD PATTERN 10
      003250  177775                         .WORD   177775
      003252  177773                         .WORD   177773
      003254  177767                         .WORD   177767
      003256  177757                         .WORD   177757
      003260  177737                         .WORD   177737
      003262  177677                         .WORD   177677
      003264  177577                         .WORD   177577
      003266  177377                         .WORD   177377
      003270  176777                         .WORD   176777
      003272  175777                         .WORD   175777
      003274  173777                         .WORD   173777
      003276  167777                         .WORD   167777
      003300  157777                         .WORD   157777
      003302  137777                         .WORD   137777
      003304  077777                         .WORD   077777

      003306  172666                DATA11:  .WORD   172666  ;STANDARD PATTERN 11
      003310  155555                         .WORD   155555
      003312  172666                        .WORD   172666
      003314  155555                        .WORD   155555
      003316  172666                        .WORD   172666
      003320  155555                        .WORD   155555
      003322  172666                        .WORD   172666
      003324  155555                        .WORD   155555
      003326  172666                        .WORD   172666
      003330  155555                        .WORD   155555
      003332  172666                        .WORD   172666
      003334  155555                        .WORD   155555
      003336  172666                        .WORD   172666
      003340  155555                        .WORD   155555
      003342  172666                        .WORD   172666
      003344  155555                        .WORD   155555

      003346  077777               DATA12:  .WORD   077777  ;STANDARD PATTERN 12
      003350  137777                        .WORD   137777
      003352  157777                        .WORD   157777
      003354  167777                        .WORD   167777
      003356  173777                        .WORD   173777
      003360  175777                        .WORD   175777
      003362  176777                        .WORD   176777
      003364  177377                        .WORD   177377
      003366  177577                        .WORD   177577
      003370  177677                        .WORD   177677
      003372  177737                        .WORD   177737
      003374  177757                        .WORD   177757
      003376  177767                        .WORD   177767
      003400  177773                        .WORD   177773
      003402  177775                        .WORD   177775
      003404  177776                        .WORD   177776

      003406  153333               DATA13:  .WORD   153333  ;STANDARD PATTERN 13
      003410  066667                        .WORD   066667
      003412  153333                        .WORD   153333
      003414  066667                        .WORD   066667
      003416  153333                        .WORD   153333
      003420  066667                        .WORD   066667
      003422  153333                        .WORD   153333
      003424  066667                        .WORD   066667
      003426  153333                        .WORD   153333
      003430  066667                        .WORD   066667
      003432  153333                        .WORD   153333
      003434  066667                        .WORD   066667
      003436  153333                        .WORD   153333
      003440  066667                        .WORD   066667
      003442  153333                        .WORD   153333
      003444  066667                        .WORD   066667

      003446  000000               DATA14:  .WORD   000000  ;STANDARD PATTERN 14
      003450  177777               ONES:    .WORD   177777  ;ALL 1'S DATA PATTERN
      003452  177777                        .WORD   177777
      003454  177777                        .WORD   177777
      003456  177777                        .WORD   177777
      003460  177777                        .WORD   177777
      003462  177777                        .WORD   177777
      003464  177777                        .WORD   177777
```

```
	003466	177777			.WORD	177777
	003470	177777			.WORD	177777
	003472	177777			.WORD	177777
	003474	177777			.WORD	177777
	003476	177777			.WORD	177777
	003500	177777			.WORD	177777
	003502	177777			.WORD	177777
	003504	177777			.WORD	177777

	003506	177777		DATA15:	.WORD	177777	;STANDARD PATTERN 15
	003510	000000			.WORD	000000
	003512	000000			.WORD	000000
	003514	000000			.WORD	000000
	003516	000000			.WORD	000000
	003520	000000			.WORD	000000
	003522	000000			.WORD	000000
	003524	000000			.WORD	000000
	003526	000000			.WORD	000000
	003530	000000			.WORD	000000
	003532	000000			.WORD	000000
	003534	000000			.WORD	000000
	003536	000000			.WORD	000000
	003540	000000			.WORD	000000
	003542	000000			.WORD	000000
	003544	000000			.WORD	000000
```

```
                                  .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


          003546                  $ERRTB:

                                  ;ERROR 1

          003546  066160                  EM1                     ;RH70 INTERRUPT OCCURRED (RMAS = 0)
          003550  070613                  DH1
          003552  071244                  DT1
          003554  000000                  0

                                  ;ERROR 2

          003556  066230                  EM2                     ;UNEXPECTED ATTENTION OCCURRED
          003560  070620                  DH2
          003562  071250                  DT2
          003564  000000                  0

                                  ;ERROR 3

          003566  066266                  EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
          003570  070674                  DH3
          003572  071266                  DT3
          003574  000000                  0

                                  ;ERROR 4

          003576  066324                  EM4                     ;MASSBUS PARITY ERROR (PAR=1)
          003600  070722                  DH4
          003602  071276                  DT4
          003604  000000                  0

                                  ;ERROR 5

          003606  066361                  EM5                     ;ADDRESS PLUG BIT CHANGED
          003610  070620                  DH2
          003612  071250                  DT2
          003614  000000                  0

                                  ;ERROR 6

          003616  066415                  EM6                     ;RH/RM DIDN'T RESPOND TO ADDRESSING
          003620  070761                  DH6
          003622  071310                  DT6
          003624  000000                  0
```

```
     43
```

```
                                        .SBTTL  PROGRAM START

003626  012737  177777  001360  START:  MOV     #-1,CHGADR          ;SET RH/RM ADDRESS CHANGE FLAG
003634  000407                          BR      START2              ;START THE PROGRAM

003636  012737  000400  001360  START1: MOV     #400,CHGADR         ;CLEAR THE RH/RM ADDRESS CHANGE FLAG
                                ;;**********************************************************************
003644  000240                  TST1:   NOP
003646  012737  000001  001212          MOV     #1,$TESTN           ;;SET TEST NUMBER IN APT MAIL BOX

003654  005227  000000          START2: INC     #0                  ;TTY LOOP, WAIT FOR INCREMENT
003660  001375                          BNE     .-4                 ;OF WORD
003662  000005                          RESET                       ;CLEAR THE WORLD


                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
003664  012706  001114                  MOV     #$CMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
003670  005026                          CLR     (R6)+               ;;CLEAR MEMORY LOCATION
003672  022706  001154                  CMP     #SWR,R6             ;;DONE?
003676  001374                          BNE     .-6                 ;;LOOP BACK IF NO
003700  012706  001100                  MOV     #STACK,SP           ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
003704  012737  032262  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
003712  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
003720  012737  035526  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
003726  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
003734  012737  033410  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
003742  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
003750  012737  176543  035112          MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
003756  012737  123456  035114          MOV     #123456,$LONUM  ;;BOTH HIGH AND LOW WORDS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
003764  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
003770  012737  004024  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
003776  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
004004  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
004012  022777  177777  175134          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
004020  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
004022  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
004024  012716  004032          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
004030  000002                          RTI
004032  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
004040  012737  000174  001156          MOV     #DISPREG,DISPLAY
004046  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

004052  005037  001214                  CLR     $PASS           ;;CLEAR PASS COUNT
004056  132737  000200  001227          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
004064  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
004066  012737  001230  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
004074                          67$:
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
004074  005227  177777                  INC     #-1             ;;FIRST TIME?
004100  001051                          BNE     68$             ;;BRANCH IF NO
004102  104401  004150                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
```

```
   004106  005737  000042                   TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
   004112  001012                           BNE     70$             ;;BRANCH IF YES
   004114  123727  001226  000001           CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
   004122  001406                           BEQ     70$             ;;BRANCH IF YES
   004124  023727  001154  000176           CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
   004132  001005                           BNE     71$             ;;BRANCH IF NO
   004134  104406                           GTSWR                   ;;GET SOFT-SWR SETTINGS
   004136  000403                           BR      71$
   004140  112737  000001  001150   70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
   004146                           71$:
   004146  000426                           BR      68$             ;;GET OVER THE ASCIZ
                                    ;;69$:  .ASCIZ  <CRLF>@CZRMUA0 - RM05/3/2 PERFORMANCE EXERCISER@<CRLF>
   004224                           68$:

                                    ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
                                    ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP

   004224  005037  001452                   CLR     XXDP            ;CLEAR 'XXDP' LOAD DEVICE STORAGE
   004230  122737  000016  000041           CMPB    #16,@#41        ;LOADED FROM AN RM05/3/2 ?
   004236  001160                           BNE     3$              ;BR IF NOT
   004240  013737  000040  001452           MOV     @#40,XXDP       ;GET DEVICE INDICATOR AND NUMBER
   004246  122737  000007  001452           CMPB    #7,XXDP         ;IS IT A VALID NUMBER ?
   004254  103002                           BHIS    1$              ;YES
   004256  105037  001452                   CLRB    XXDP            ;NO, DEFAULT TO DRIVE 0
   004262  005737  000042           1$:     TST     @#42            ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
   004266  001425                           BEQ     2$              ;BR IF NEITHER
   004270  104401  004276                   TYPE    ,73$            ;;TYPE ASCIZ STRING
   004274  000412                           BR      72$             ;;GET OVER THE ASCIZ
                                    ;;73$:  .ASCIZ  <CRLF>/NOT TESTING DRIVE /
   004322                           72$:
   004322  005046                           CLR     -(SP)           ;CLEAR WORD ON STACK
   004324  113716  001452                   MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
   004330  104403                           TYPOS                   ;TYPE THE ADDRESS
   004332     001                           .BYTE   1               ;ONLY 1 CHARACTER
   004333     000                           .BYTE   0               ;SUPRESS LEADING ZEROS
   004334  104401  001203                   TYPE    ,$CRLF          ;CR-LF
   004340  000517                           BR      3$              ;GET NUMBER OF DRIVES

   004342  005227  177777           2$:     INC     #-1             ;FIRST TIME THRU HERE ?
   004346  001114                           BNE     3$              ;NO
   004350  104401  004356                   TYPE    ,75$            ;;TYPE ASCIZ STRING
   004354  000410                           BR      74$             ;;GET OVER THE ASCIZ
                                    ;;75$:  .ASCIZ  <CRLF>/TO TEST DRIVE /
   004376                           74$:
   004376  005046                           CLR     -(SP)           ;CLEAR WORD ON STACK
   004400  113716  001452                   MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
   004404  104403                           TYPOS                   ;TYPE DRIVE ADDRESS
   004406     001                           .BYTE   1               ;ONLY 1 CHARACTER
   004407     000                           .BYTE   0               ;SUPRESS LEADING ZEROS
   004410  104401  004416                   TYPE    ,77$            ;;TYPE ASCIZ STRING
   004414  000431                           BR      76$             ;;GET OVER THE ASCIZ
                                    ;;77$:  .ASCIZ  /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
   004500                           76$:
   004500  104401  004506                   TYPE    ,78$            ;;TYPE ASCIZ STRING
   004504  000435                           BR      3$              ;;GET OVER THE ASCIZ
                                    ;;78$:  .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
   004600                           3$:
```

```
004600  004737  031604                   JSR     PC,$TKINT          ;TURN ON THE KEYBOARD INTERRUPT
                                 .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
004604  005737  000042                   TST     @#42               ;;ARE WE RUNNING UNDER XXDP/ACT?
004610  001012                           BNE     79$                ;;BRANCH IF YES
004612  123727  001226  000001           CMPB    $ENV,#1            ;;ARE WE RUNNING UNDER APT?
004620  001406                           BEQ     79$                ;;BRANCH IF YES
004622  023727  001154  000176           CMP     SWR,#SWREG         ;;SOFTWARE SWITCH REG SELECTED?
004630  001005                           BNE     80$                ;;BRANCH IF NO
004632  104406                           GTSWR                      ;;GET SOFT-SWR SETTINGS
004634  000403                           BR      80$
004636  112737  000001  001150   79$:    MOVB    #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
004644                           80$:
004644  105737  001226                   TSTB    $ENV               ;RUN UNDER APT MODE
004650  001423                           BEQ     4$                 ;NO,DO NOT BOTHER
004652  105737  001256                   TSTB    $VECT1             ;NEW VECTOR ?
004656  001403                           BEQ     .+10               ;NOT LOAD IF = 0
004660  113737  001256  001274           MOVB    $VECT1,$RMVEC      ;NEW VECTOR
004666  005737  001262                   TST     $BASE              ;NEW BASE ADDRESS ?
004672  001403                           BEQ     .+10               ;NO
004674  013737  001262  001272           MOV     $BASE,$RMADR       ;NEW BASE ADDRESS
004702  013737  001272  035762           MOV     $RMADR,RMADR       ;LOAD ADDRESS INTO DRIVER
004710  013737  001274  035764           MOV     $RMVEC,RMVEC       ;LOAD VECTOR INTO DRIVER
004716  000420                           BR      5$

004720  105737  001150           4$:     TSTB    $AUTOB             ;AUTO MODE ?
004724  001015                           BNE     5$                 ;YES
004726  004737  076500                   JSR     PC,BUSADR          ;CHECK RH/RM BUS ADDRESS
004732  013737  001272  035762           MOV     $RMADR,RMADR       ;RH/RM ADDRESS
004740  013737  001274  035764           MOV     $RMVEC,RMVEC       ;RH/RM VECTOR ADDRESS
004746  005227  177777                   INC     #-1                ;FIRST TIME THRU HERE ?
004752  001002                           BNE     5$                 ;BR IF NO
004754  004737  076160                   JSR     PC,OPRDAT          ;GET THE DATE AND OPERATOR ID
004760  005037  001316           5$:     CLR     STATIN             ;CLEAR PERFORMANCE SUMMARY TYPEOUT FLAG
004764  012705  001526                   MOV     #ORDERQ,R5         ;START OF AREA TO CLEAR
004770  005025                   6$:     CLR     (R5)+
004772  022705  002106                   CMP     #BLKADR,R5         ;LOOK FOR END OF CLEAR AREA
004776  001374                           BNE     6$                 ;BR IF NOT FINISHED
005000  012706  001100                   MOV     #STACK,SP          ;SETUP THE STACK POINTER
005004  005037  177776                   CLR     PS                 ;CLEAR THE PROCESSOR STATUS WORD
005010  013737  001314  001374           MOV     HZ,SIXTEE          ;1/60 TH OR 1/50 TH SECOND COUNTER VALUE
005016  005037  001366                   CLR     HOUR               ;CLEAR THE HOUR'S COUNTER
005022  005037  001370                   CLR     MINUTE             ;CLEAR THE MINUTE'S COUNTER
005026  005037  001372                   CLR     SECOND             ;CLEAR THE SECOND'S COUNTER
005032  005037  001474                   CLR     INTRVL+2           ;CLEAR INTERVAL COUNTER
005036  005037  001320                   CLR     PACK               ;CLEAR THE 'R' OR 'W' COMMAND FLAG
005042  005037  001362                   CLR     CFLAG              ;CLEAR THE 'CONTROL C' FLAG
005046  042737  170000  001470           BIC     #170000,MAXER      ;MAKE SURE ERROR LIMITS ARE NOT TOO HIGH

                                 ;ROUTINE TO DETERMINE BUFFER AREA SIZE

005054  005227  177777           SIZMEM: INC     #-1                ;FIRST TIME THRU HERE ?
005060  001005                           BNE     1$                 ;BR IF NO
005062  004737  076346                   JSR     PC,$SIZE           ;SEE HOW MUCH MEMORY ON SYSTEM
005066  013737  076476  001356           MOV     $LSTAD,LSTAD       ;SAVE THE LAST ADDRESS
005074  012737  000001  001704   1$:     MOV     #1,BUFTBL          ;LOAD NUMBER OF BUFFERS
005102  012737  077752  001706           MOV     #ENDPGM,BUFTBL+2  ;STARTING ADDRESS OF BUFFER
005110  013737  001356  001710           MOV     LSTAD,BUFTBL+4     ;LAST ADDR TO BUFFER ALLOCATION TABLE
```

```
005116  023727  001356  160000          CMP     LSTAD,#160000     ;OVER 28K ?
005124  101403                          BLOS    2$                ;NO
005126  012737  160000  001710          MOV     #160000,BUFTBL+4        ;XXDP MAX MEMORY 28K
005134  162737  077752  001710  2$:     SUB     #ENDPGM,BUFTBL+4  ;SUBTRACT PROGRAM SPACE
005142  000241                          CLC                       ;CLEAR THE 'C' BIT
005144  006037  001710                  ROR     BUFTBL+4          ;CONVERT TO WORD COUNT
005150  162737  000144  001710          SUB     #100.,BUFTBL+4    ;SAVE ROOM FOR THE 'ABS' LOADER
005156  005737  000042                  TST     42                ;LOAD FROM XXDP OR OTHER MONITOR ?
005162  001403                          BEQ     3$                ;BR IF LOADED BY PAPER TAPE
005164  162737  002570  001710          SUB     #1400.,BUFTBL+4   ;SUBTRACT 'XXDP' LOADER SIZE
005172  005737  001466          3$:     TST     MAXDL             ;VALUE IN 'MAXDL' ?
005176  001003                          BNE     4$                ;BR IF VALUE IS
005200  012737  020000  001466          MOV     #8192.,MAXDL      ;ASSUME FULL TRACK   MAXIMUM
005206  023737  001466  001710  4$:     CMP     MAXDL,BUFTBL+4    ;IS THAT TOO LARGE ?
005214  103403                          BLO     5$                ;BR IF NOT
005216  013737  001710  001466          MOV     BUFTBL+4,MAXDL    ;USE MAX AVAIL MEMORY AS MAX BUFFER SIZE
005224  013737  001710  075122  5$:     MOV     BUFTBL+4,PARLST+2  ;VALUE FOR THE PARAMETER TABLE

                                ;SEE IF THE OPERATOR WANTS TO CHANGE ANY PARAMETERS

005232  105737  001150          LKPAR:  TSTB    $AUTOB                  ;'XXDP' CHAIN MODE OR 'ACT11' OPERATION ?
005236  001044                          BNE     SETVEC                  ;BR IF YES
005240  022737  007070  001266          CMP     #7070,$CDW1             ;FROM THE POWER FAIL RT. ?
005246  001440                          BEQ     SETVEC                  ;BRANCH IF SO
005250  105737  001226                  TSTB    $ENV                    ;APT STAND ALONE MODE ?
005254  001035                          BNE     SETVEC                  ;NO
005256  104401  075222                  TYPE    ,ASKPAR                 ;ASK FOR PARMETERS
005262  104410                          RDCHR                           ;READ THE ENTRY
005264  012637  001174                  MOV     (SP)+,$TMPO             ;SAVE RESPONSE
005270  122737  000131  001174          CMPB    #'Y,$TMPO               ;WAS IT A YES RESPONSE ?
005276  001405                          BEQ     1$                      ;BR IF YES
005300  104401  074244                  TYPE    ,N                      ;TYPE 'N'
005304  104401  001203                  TYPE    ,$CRLF                  ;CR-LF
005310  000417                          BR      SETVEC
005312  104401  001174          1$:     TYPE    ,$TMPO                  ;TYPE 'Y'
005316  104401  001203                  TYPE    ,$CRLF                  ;CR-LF

005322  012703  075120          ENTPR:  MOV     #PARLST,R3              ;PARAMETER TABLE ADDRESS
005326  004737  027702                  JSR     PC,PARENT               ;GET THE PARAMETER ENTRY
005332  023727  001466  000004          CMP     MAXDL,#4                ;IS THE 'MAXDL' VALUE OK ?
005340  103003                          BHIS    SETVEC                  ;BR IF IT IS
005342  012737  000004  001466          MOV     #4,MAXDL                ;SET 'MAXDL' TO THE MINIMUM VALUE

                                ;DISPLAY DRIVE STATUS AND SET UP THE OTHER SYSTEM DEVICES THAT
                                ;THE PROGRAM WILL USE

005350  004737  023326          SETVEC: JSR     PC,CKCLK                ;START THE CLOCK
005354  004737  036000                  JSR     PC,RMINIT               ;INITIALIZE THE RM DRIVER
005360  012737  177777  035722          MOV     #-1,SAVEFG              ;SET THE SAVE REGISTERS FLAG
005366  005227  177777                  INC     #-1                     ;FIRST TIME THRU ?
005372  001404                          BEQ     1$                      ;BR IF YES
005374  032777  000004  173552          BIT     #SW02,@SWR              ;TYPEOUT THE DRIVE STATUS TABLE ?
005402  001104                          BNE     12$                     ;BR IF NOT
005404  005004                  1$:     CLR     R4                      ;DRIVE TABLE POINTER
005406  104401  073535                  TYPE    ,SYSTAT                 ;TYPE STATUS HEADING
005412  104401  001203          2$:     TYPE    ,$CRLF                  ;CR-LF
005416  010446                          MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
```

```
                                                              ;;TYPE DRIVE NUMBER
    005420  104403                            TYPOS           ;;GO TYPE--OCTAL ASCII
    005422     002                            .BYTE   2       ;;TYPE 2 DIGIT(S)
    005423     000                            .BYTE   0       ;;SUPPRESS LEADING ZEROS
    005424  104401  073266                    TYPE    ,BLNKS4         ;TYPE 4 BLANKS
    005430  105764  035634                    TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
    005434  100416                            BMI     5$              ;BR IF UNSAFE
    005436  001020                            BNE     6$              ;BR IF ONLINE

    005440  105764  035644                    TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
    005444  001404                            BEQ     3$              ;BR IF NONEXISTENT
    005446  100006                            BPL     4$              ;BR IF OFFLINE
    005450  104401  073430                    TYPE    ,NOTRM          ;DRIVE NOT AN RM05/3/2
    005454  000451                            BR      11$             ;CHECK NEXT DRIVE

    005456  104401  073451            3$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
    005462  000446                            BR      11$             ;CHECK NEXT DRIVE

    005464  104401  073337            4$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
    005470  000416                            BR      8$              ;PRINT DRIVE TYPE

    005472  104401  073505            5$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
    005476  000413                            BR      8$              ;PRINT DRIVE TYPE

    005500  005737  001452            6$:     TST     XXDP            ;LOADED FROM THIS DEVICE ?
    005504  001406                            BEQ     7$              ;BR IF NO
    005506  123704  001452                    CMPB    XXDP,R4         ;LOADED FROM THIS DRIVE ?
    005512  001003                            BNE     7$              ;BR IF NO
    005514  104401  073515                    TYPE    ,LODEV          ;DRIVE IS LOAD DEVICE
    005520  000427                            BR      11$
    005522  104401  073350            7$:     TYPE    ,UNTON          ;DRIVE ONLINE
    005526  104401  073270            8$:     TYPE    ,BLNKS2         ;TYPE 2 BLANKS
    005532  012737  073560  005576            MOV     #$RM03,10$      ;ADDRESS OF RM03 MESSAGE
    005540  122764  000004  035644            CMPB    #4,DRVTYP(R4)   ;IS DEVICE AN RM03 ?
    005546  001412                            BEQ     9$              ;BR IF YES
    005550  012737  073553  005576            MOV     #$RM02,10$      ;ADDRESS OF RM02 MESSAGE
    005556  122764  000005  035644            CMPB    #5,DRVTYP(R4)   ;IS DEVICE AN RM02 ?
    005564  001403                            BEQ     9$              ;BR IF YES
    005566  012737  073565  005576            MOV     #$RM05,10$      ;ADDRESS OF RM05 MESSAGE

    005574  104401                    9$:     TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
    005576  000000                    10$:    .WORD   0               ;MESSAGE ADDRESS HERE

    005600  005204                    11$:    INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
    005602  020427  000010                    CMP     R4,#8.          ;FINISHED ?
    005606  001301                            BNE     2$              ;BR IF NOT
    005610  104401  001203                    TYPE    ,$CRLF          ;CR-LF
    005614                            12$:

                                      ;SETUP IF 'XXDP' OR 'ACT11' OPERATION

    005614  105737  001150            MONTR:  TSTB    $AUTOB          ;'XXDP' CHAIN MODE OR 'ACT11' AUTO ACCEPT
    005620  001407                            BEQ     1$              ;BR IF NEITHER
    005622  005737  001360                    TST     CHGADR          ;200 START ?
    005626  003004                            BGT     1$              ;YES
    005630  004737  025260                    JSR     PC,ASGN2        ;ASSIGN DRIVES
    005634  104401  001203                    TYPE    ,$CRLF          ;CR-LF
```

```
005640  004737  031604          1$:     JSR     PC,$TKINT         ;INITIALIZE THE KEYBOARD INTERRUPT HANDLER

                                ;FORCE TO TEST PACK FOR 200 START
005644  012737  002740  001454  FOWT1:  MOV     #2740,ENDCON      ;INITIAL XTRANS COUNT
005652  012737  005455  001456          MOV     #5455,ENDCON+2    ;
005660  105737  001226                  TSTB    $ENV              ;APT SCRIPT MODE, THEN MAKE IT RUN 2 MIN.
005664  001412                          BEQ     1$                ;NO
005666  012737  000001  001504          MOV     #1,RATIO          ;SPEED UP TEST
005674  012737  077777  001454          MOV     #77777,ENDCON     ;SET LSW OF ENDING VALUE
005702  012737  000027  001456          MOV     #27,ENDCON+2      ;SET MSW OF ENDING VALUE
005710  000403                          BR      FOWT2             ;START TO WRITE AND TEST
005712  005737  001360          1$:     TST     CHGADR            ;START AT 200 ?
005716  003464                          BLE     FOWT3             ;NO

005720  005001                  FOWT2:  CLR     R1                ;DRIVE #
005722  005002                          CLR     R2                ;AVAIL TABLE INDEX
005724  005003                          CLR     R3                ;DRIVE# * 2
005726  005737  001452          1$:     TST     XXDP              ;LOADED FROM THIS DEVICE ?
005732  001403                          BEQ     2$                ;BR IF NO
005734  123701  001452                  CMPB    XXDP,R1           ;LOADED FROM THIS DRIVE ?
005740  001442                          BEQ     4$                ;BR IF YES
005742  105761  035634          2$:     TSTB    DRVSTA(R1)        ;DRIVE ON LINE ?
005746  003437                          BLE     4$                ;NO
005750  016300  002106                  MOV     BLKADR(R3),R0     ;LOAD DPB ADDRESS
005754  004737  026174                  JSR     PC,CLRDPB         ;CLEAR DPB BLOCK
005760  004537  026700                  JSR     R5,GETADR         ;RETRIEVE BAD SPOT FILE
005764  000430                          BR      4$                ;UNSUCCESSFUL !
005766  010062  001574                  MOV     R0,NEWUNT(R2)     ;LOAD DPB ADDRESS TO ABAIL QUEUE
005772  112760  177776  000026          MOVB    #-2,$PACK(R0)     ;WRITE PATTERN THEN TEST
006000  013760  001444  000106          MOV     CYLIMT,MAXCYL(R0) ;UP CYLINDER LIMIT
006006  004737  026546                  JSR     PC,GETLMT         ;GET ADDRESS LIMITS
006012  013760  001450  000112          MOV     TRKLMT,MAXTRK(R0) ;UPPER TRACK LIMIT
006020  013760  001446  000116          MOV     SECLMT,MAXSEC(R0) ;UPPER SECTOR LIMIT
006026  005060  000110                  CLR     MINCYL(R0)        ;CLEAR LOWER LIMIT
006032  005060  000114                  CLR     MINTRK(R0)        ;CLEAR LOWER LIMIT
006036  005060  000120                  CLR     MINSEC(R0)        ;CLEAR LOWER LIMIT
006042  004737  020374                  JSR     PC,WRTPK          ;SET UP PARAMETERS

006046  022322                  4$:     CMP     (R3)+,(R2)+       ;INCREMENT INDEX
006050  005201                          INC     R1                ;NEXT DRIVE
006052  020127  000007                  CMP     R1,#7             ;ALL DRIVE ASSIGN ?
006056  003723                          BLE     1$                ;NO
006060  005037  001320                  CLR     PACK              ;TEST PACK FLAG
006064  000137  006260                  JMP     MAIN1             ;JUMP TO WAIT PARAMETER AND BUFFER LOOP

006070  104401  074610          FOWT3:  TYPE    ,INTDON           ;TYPE 'INITIALIZE COMPLETE'
006074  000137  006260                  JMP     MAIN1             ;START THE PROGRAM

```

```
                                   .SBTTL  MAIN PROGRAM

 006100  005737  001362            MAIN:   TST     CFLAG           ;KEYBOARD INTERRUPTED ?
 006104  001407                            BEQ     3$              ;BRANCH IF NOT
 006106  005737  001526            1$:     TST     ORDERQ          ;DON'T DEASSIGN ANY DRIVE,IF SYSTEM NOT IDLE
 006112  001402                            BEQ     2$              ;
 006114  000137  006676                    JMP     IDLE            ;LET ALL DRIVE FINISH ORDER
 006120  004737  024624            2$:     JSR     PC,KSR          ;SERVICE THE KEYBOARD
 006124  000240                    3$:     NOP                     ;EXIT

 006126  012703  000010            MAINDA: MOV     #8.,R3          ;DRIVE COUNTER
 006132  012705  001552                    MOV     #DUNIT,R5       ;ADDRESS OF 'DROP DRIVE' TABLE
 006136  005715                    1$:     TST     (R5)            ;SEE IF ENTRY AT PRESENT POSITION
 006140  001011                            BNE     3$              ;BR IF THERE IS ONE
 006142  062705  000002            2$:     ADD     #2,R5           ;INCREMENT TO NEXT TABLE POSITION
 006146  005303                            DEC     R3              ;DECREMENT DRIVE COUNTER
 006150  001372                            BNE     1$              ;BR IF MORE TO CHECK
 006152  105737  001550                    TSTB    ASNLST          ;ANY DRIVES ACTIVE ?
 006156  001040                            BNE     MAIN1           ;BR IF YES
 006160  000137  030526                    JMP     $GET42          ;GIVE CONTROL TO MONITOR

 006164  012701  001616            3$:     MOV     #AVAIL,R1       ;ADDRESS OF 'AVAILABLE DRIVES' TABLE
 006170  005711                    4$:     TST     (R1)            ;SEE IF AT END OF TABLE
 006172  001405                            BEQ     5$              ;BR IF AT END: GO CHECK 'WAIT' TABLE
 006174  021115                            CMP     (R1),(R5)       ;IS DRIVE IN 'AVAIL' THE ONE TO BE DROPPED
 006176  001414                            BEQ     7$              ;BR IF YES
 006200  062701  000002                    ADD     #2,R1           ;INCREMENT 'AVAIL' TABLE ADDRESS
 006204  000771                            BR      4$              ;CONTINUE LOOKING
 006206  012701  001640            5$:     MOV     #WAIT,R1        ;MOVE THE ADDRESS OF THE BUFFER WAIT TABLE
 006212  005711                    6$:     TST     (R1)            ;AT THE END OF THE 'WAIT' TABLE ?
 006214  001752                            BEQ     2$              ;BR IF YES: SEE IF ANY MORE 'DROP' REQUESTS
 006216  021115                            CMP     (R1),(R5)       ;DRIVE IN THE 'WAIT' TABLE ?
 006220  001403                            BEQ     7$              ;BR IF IT IS
 006222  062701  000002                    ADD     #2,R1           ;INCREMENT 'WAIT' TABLE ADDRESS
 006226  000771                            BR      6$              ;CONTINUE LOOK THROUGH THE 'WAIT' TABLE
 006230  011100                    7$:     MOV     (R1),R0         ;PUT THE DRIVE'S BLOCK ADDRESS IN R0
 006232  104401  001203                    TYPE    ,$CRLF          ;CR-LF
 006236  104401  074114                    TYPE    ,DEASSG         ;TYPE 'DRIVE DEASSIGNED'
 006242  004737  023616                    JSR     PC,SUMARY       ;TYPE THE DRIVE'S PERFORMANCE SUMMARY
 006246  005015                            CLR     (R5)            ;CLEAR THE 'DROP DRIVE' TABLE ENTRY
 006250  005011                            CLR     (R1)            ;REMOVE THE DRIVE FROM THE 'AVAIL' OR 'WAIT' TABLE
 006252  004737  020356                    JSR     PC,CMPRES       ;COMPRESS THE RESPECTIVE TABLE
 006256  000731                            BR      2$              ;SEE IF ANY MORE DRIVES

                                   ;LOOK FOR DRIVES TO BE ASSIGNED

 006260  012703  000010            MAIN1:  MOV     #8.,R3          ;DRIVE COUNT
 006264  005002                            CLR     R2              ;'AVAIL' INDEX
 006266  005004                            CLR     R4              ;ASSIGN LIST INDEX
 006270  005005                            CLR     R5              ;NEW DRIVE INDEX
 006272  005765  001574            1$:     TST     NEWUNT(R5)      ;NEW DRIVE IN THIS POSITION
 006276  001006                            BNE     3$              ;BR IF THERE IS
 006300  062705  000002            2$:     ADD     #2,R5           ;INCREMENT R5
 006304  005204                            INC     R4              ;INCREMENT ASSIGN INDEX
 006306  005303                            DEC     R3              ;DECREMENT DRIVE COUNT
 006310  001370                            BNE     1$              ;BR IF MORE DRIVES
 006312  000432                            BR      MAIN2           ;START OPERATIONS FOR THE AVAILABLE DRIVES
```

```
006314  104401  001203          3$:     TYPE    ,$CRLF                  ;CR-LF
006320  104401  073331                  TYPE    ,UNTMSG                 ;'DRIVE'
006324  010446                          MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
006326  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
006330     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
006331     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
006332  104401  074162                  TYPE    ,ASGND                  ;'STARTED'
006336  005762  001616          4$:     TST     AVAIL(R2)               ;AT END OF AVAILABLE TABLE
006342  001403                          BEQ     5$                      ;BR IF YES
006344  062702  000002                  ADD     #2,R2                   ;INCREMENT AVAILABLE TABLE INDEX
006350  000772                          BR      4$                      ;CONTINUE LOOKING FOR END OF TABLE
006352  016562  001574  001616  5$:     MOV     NEWUNT(R5),AVAIL(R2)    ;MOVE ADDR OF DRIVE INTO AVAIL LST
006360  005065  001574                  CLR     NEWUNT(R5)              ;TAKE DRIVE OUT OF NEW DRIVE TABLE
006364  156437  035750  001550          BISB    ATABIT(R4),ASNLST ;SET DRIVE ASSIGNED INDICATOR
006372  062702  000002                  ADD     #2,R2                   ;INCREMENT AVAILABLE TABLE POINTER
006376  000740                          BR      2$                      ;LOOK FOR MORE DRIVES

                                ;GET PARAMETERS, BUFFER SPACE, AND START ORDERS FOR DRIVES IN
                                ;THE 'AVAILABLE' QUEUE

006400                          MAIN2:                                  ;SET UP THE WAIT QUEUE FOR
                                                                        ;ALL THE DRIVES THAT WAITING
                                                                        ;FOR BUFFER IN THREE TIMES OF
                                                                        ;SEARCHING
006400  005002                          CLR     R2                      ;START FROM THE FIRST LOCATION
006402  005762  001640          1$:     TST     WAIT(R2)                ;WAIT FOR THE BUFFER ?
006406  001434                          BEQ     MAIN3                   ;EXIT IF NONE
006410  016200  001640                  MOV     WAIT(R2),R0             ;LOAD R0 WITH THE DPB ADDRESS
006414  005046                          CLR     -(SP)                   ;CLEAR THE STACK FOR BUFFER REQ
006416  004737  016312                  JSR     PC,GETBUF               ;CALL TO GET THE BUFFER RT.
006422  012660  000006                  MOV     (SP)+,$BUF(R0)          ;IF 0,BUFFER IS STILL NOT AVAILABLE
006426  001421                          BEQ     2$                      ;BRANCH IF NO BUFFER AVAILABLE
006430  005060  000072                  CLR     $FAIR(R0)               ;CLEAR THE FAIR FLAG
006434  004737  016676                  JSR     PC,FILBUF               ;FILL THE BUFFER
006440  004737  017024                  JSR     PC,GODRIV               ;SET COMMAND AND GO
006444  012705  001526                  MOV     #ORDERQ,R5              ;PUT THE WAIT QUEUE INTO ORDER QUEUE
006450  005725                          TST     (R5)+                   ;QUEUE AVAILABLE ?
006452  001376                          BNE     .-2                     ;BRANCH IF NOT
006454  010045                          MOV     R0,-(R5)                ;LOAD THE DPB ADDRESS INTO THE ORDER QUEUE
006456  012701  001640                  MOV     #WAIT,R1                ;REMOVE THE QUEUE FROM THE WAITING QUEUE
006462  060201                          ADD     R2,R1                   ;OFFSET THE QUEUE POSITION
006464  004737  020356                  JSR     PC,CMPRES               ;COMPRESS THE QUEUE
006470  000744                          BR      1$                      ;BRANCH IF DONE
006472  062702  000002          2$:     ADD     #2,R2                   ;CHECK THE NEXT QUEUE
006476  000741                          BR      1$                      ;LOOPING BACK

006500  005737  001526          MAIN3:  TST     ORDERQ                  ;OUTSTANDING BUFFER REQUESTS
006504  001074                          BNE     IDLE                    ;BR IF THERE ARE
006506  005002                          CLR     R2                      ;CLEAR DRIVE TABLE POINTER
006510  005762  001616          1$:     TST     AVAIL(R2)               ;ANY DRIVES WAITING FOR PARAMETERS
006514  001002                          BNE     .+6                     ;BRANCH IF ANY
006516  000137  006676                  JMP     IDLE                    ;BRANCH IF NONE
006522  016200  001616                  MOV     AVAIL(R2),R0            ;CONTROL BLOCK ADDR IN R0
006526  005760  000104                  TST     $NEXT(R0)               ;PARAMETERS BEEN SELECTED ?
006532  001010                          BNE     6$                      ;BR IF THEY HAVE
006534  105760  000026                  TSTB    $PACK(R0)               ;'R' OR 'W' COMMAND FOR THE DRIVE ?
```

```
006540  001403                            BEQ     5$                ;BR IF NOT
006542  004737  020374                    JSR     PC,WRTPK          ;GET DATA PACK PARAMETERS
006546  000404                            BR      7$                ;GET THE BUFFER
006550  004737  017102            5$:     JSR     PC,SELPAR         ;SELECT THE PARAMETERS
006554  004737  020064            6$:     JSR     PC,GETPAR         ;LOAD NEW PARAMETERS
006560  005046                    7$:     CLR     -(SP)             ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
006562  004737  016312                    JSR     PC,GETBUF         ;GET BUFFER
006566  012660  000006                    MOV     (SP)+,$BUF(R0)    ;MOVE BUFFER ADDR TO DPB
006572  001414                            BEQ     8$                ;BR IF '0' ADDR (NO BUFFER)
006574  004737  016676                    JSR     PC,FILBUF         ;FILL THE BUFFER
006600  005060  000072                    CLR     $FAIR(R0)         ;CLEAR THE 'FAIRNESS' COUNT
006604  004737  017024                    JSR     PC,GODRIV         ;PUT CURRENT DPB IN DRIVER
006610  012705  001526                    MOV     #ORDERQ,R5        ;ADDRESS OF ORDER QUEUE IN R5
006614  005725                            TST     (R5)+             ;END OF QUEUE ?
006616  001376                            BNE     .-2               ;BR IF NOT
006620  010045                            MOV     R0,-(R5)          ;PUT BLOCK ADDRESS INTO QUEUE
006622  000417                            BR      10$               ;CONTINUE LOOKING
006624                            8$:
006624  005260  000072                    INC     $FAIR(R0)         ;INCREMENT THE FAIR COUNT
006630  022760  000003  000072            CMP     #3,$FAIR(R0)      ;THREE TIMES,BUFFER IS NOT AVAILABLE?
006636  101006                            BHI     9$                ;BRANCH IF NOT OVER THREE TIMES
006640  012705  001640                    MOV     #WAIT,R5          ;LOAD INTO THE WAIT QUEUE
006644  005725                            TST     (R5)+             ;AN AVAILABLE LOCATION ?
006646  001376                            BNE     .-2               ;BRANCH IF NOT
006650  010045                            MOV     R0,-(R5)          ;LOAD INTO WAIT QUE
006652  000403                            BR      10$               ;REMOVE THE DPB FROM AVAILABLE QUE
006654                            9$:
006654  062702  000002                    ADD     #2,R2             ;INCREMENT INDEX
006660  000713                            BR      1$                ;BRANCH BACK TO FIRE NEXT DRIVE
006662  012701  001616            10$:    MOV     #AVAIL,R1         ;'AVAILABLE' TABLE ADDRESS
006666  060201                            ADD     R2,R1             ;FORM ADDRESS OF LAST ENTRY
006670  004737  020356                    JSR     PC,CMPRES         ;COMPRESS THE TABLE
006674  000705                            BR      1$                ;CONTINUE LOOKING

                                  ;GET BUFFER ASSIGNMENTS FOR DRIVES IN THE 'BUFFER WAIT' QUEUE


                                  ;WAIT FOR AN ORDER TO FINISH

006676  012701  001526            IDLE:   MOV     #ORDERQ,R1        ;ADDRESS OF THE ORDER QUEUE IN R1
006702  012100                    1$:     MOV     (R1)+,R0          ;PUT BLOCK ADDRESS INTO R0
006704  001433                            BEQ     IDLE1             ;BR IF END OF QUEUE
006706  005760  000016            16$:    TST     $STATUS(R0)       ;SEE IF DRIVE FINISHED
006712  001775                            BEQ     16$               ;BR IF DRIVE NOT FINISHED
006714  162701  000002                    SUB     #2,R1             ;CORRECT THE QUEUE POINTER
006720  010146                            MOV     R1,-(SP)          ;SAVE THE QUEUE ADDRESS
006722  004737  016154                    JSR     PC,STATIS         ;ACCUMULATE STATISTICS FOR DRIVE IN R0
006726  000240                            NOP                       ;DEBUGGING AID
006730  004737  007162                    JSR     PC,PROCES         ;PROCESS END OF ORDER
006734  005037  001364                    CLR     BADSEC            ;CLEAR THE BAD TRK/SEC ERROR INDICATOR
006740  004737  030152                    JSR     PC,ABNRML         ;SEE IF ANY DRIVES HAVE TOO MANY ERRORS
006744  004737  030200                    JSR     PC,$EOP           ;IS IT END OF PASS ?
006750  012601                            MOV     (SP)+,R1          ;RESTORE THE ORDER TABLE INDEX
006752  012705  001616                    MOV     #AVAIL,R5         ;FIND THE END OF THE 'AVAILABLE' TABLE
006756  005725                    2$:     TST     (R5)+             ;END OF THE TABLE ?
006760  001376                            BNE     2$                ;BR IF NOT AT END OF LIST
006762  011145                            MOV     (R1),-(R5)        ;MOVE THE BLOCK ADDRESS INTO THE TABLE
```

```
006764  004737  020356                   JSR     PC,CMPRES          ;COMPRESS THE ORDER QUEUE
006770  004737  016446                   JSR     PC,RELBUF          ;RESTORE BUFFER
006774                          IDLE1:
006774  032777  000004  172152  1$:      BIT     #SW02,@SWR         ;TYPE PERFORMANCE SUMMARY
007002  001014                           BNE     2$                 ;BR IF NOT
007004  005737  001316                   TST     STATIN             ;TIME TO TYPE THE PERFORMANCE SUMMARY ?
007010  001411                           BEQ     2$                 ;BR IF NOT
007012  005037  001316                   CLR     STATIN             ;CLEAR THE INDICATOR
007016  005737  001550                   TST     ASNLST             ;ANY DRIVES ASSIGNED ?
007022  001404                           BEQ     2$                 ;BR IF NO
007024  104401  073572                   TYPE    ,REPHD             ;TYPE PERFORMANCE REPORT HEADING
007030  004737  023530                   JSR     PC,STATPR          ;TYPE THE SUMMARY
007034  000137  006100          2$:      JMP     MAIN               ;CONTINUE THE LOOP

                                ;SETUP TO REFORMAT AN ERROR SECTOR

007040  032777  000001  172106  REFMT:   BIT     #SW0,@SWR          ;READ ONLY SWITCH SET ?
007046  001044                           BNE     REFMTX             ;BR IF IT IS
007050  032777  000200  172076           BIT     #SW7,@SWR          ;SWITCH 7 SET ?
007056  001040                           BNE     REFMTX             ;BR IF IT IS
007060  005737  001500                   TST     FORMAT             ;WRITE HEADER & DATA ORDERS ALLOWED ?
007064  001435                           BEQ     REFMTX             ;BR IF NOT
007066  022760  001465  002152           CMP     #821.,$RMDC(R0)    ;LEGAL VALUE ?
007074  101431                           BLOS    REFMTX             ;NO, NOT FORMAT
007076  004737  023176                   JSR     PC,READDR          ;GET CORRECTED SECTOR-TRACK ADDRESSES
007102  012660  000100                   MOV     (SP)+,$NCYL(R0)         ;CYLINDER
007106  112660  000077                   MOVB    (SP)+,$NTRK(R0)    ;TRACK ADDR TO DPB
007112  112660  000076                   MOVB    (SP)+,$NSEC(R0)    ;SECTOR ADDR TO DPB
007116  012760  000402  000102           MOV     #258.,$NWRDL(R0)   ;WORD COUNT FOR FORMAT
007124  112760  000003  000074  1$:      MOVB    #3,$NCODE(R0)      ;COMMAND CODE
007132  004537  017656                   JSR     R5,CHKADR          ;AVOID REFORMAT BAD SPOT
007136  000401                           BR      2$                 ;BRANCH IF NOT ON BAD SPOT
007140  000407                           BR      REFMTX             ;BRANCH IF ON BAD SPOT
007142  004737  020022          2$:      JSR     PC,GETPAT          ;GET A PATTERN
007146  110560  000075                   MOVB    R5,$NPATC(R0)      ;PATTERN CODE TO CONTROL BLOCK
007152  012760  177777  000104           MOV     #-1,$NEXT(R0)      ;SET PARAMETERS SELECTED INDICATOR
007160  000207                  REFMTX:  RTS     PC                 ;RETURN

                                ;PROCESS THE ORDER TERMINATION

007162  111037  001346          PROCES:  MOVB    (R0),UNIT          ;DRIVE NUMBER FOR ANY ERROR MESSAGES
007166  005760  000016                   TST     $STATUS(R0)        ;SEE IF DRIVER SIGNALED AN ERROR
007172  100427                           BMI     ERPROC             ;BR IF ERROR
007174  032760  100000  002116           BIT     #BIT15,$RMCS1(R0)  ;SEE IF 'SC' SET
007202  001410                           BEQ     1$                 ;BR IF NOT SET
007204  032760  040000  002116           BIT     #BIT14,$RMCS1(R0)  ;SEE IF 'TRE' SET
007212  001017                           BNE     ERPROC             ;BR IF SET
007214  032760  040000  002130           BIT     #BIT14,$RMDS(R0)   ;SEE IF 'ERR' SET
007222  001013                           BNE     ERPROC             ;BR IF SET
007224  004737  013306          1$:      JSR     PC,CKERR           ;NO ERROR, CHECK ERROR BITS ANYWAY
007230  004737  013400                   JSR     PC,CKBUS           ;NO ERROR, CHECK BUS ADDR & WC
007234  032777  000002  171712           BIT     #SW01,@SWR         ;DATA COMPARE ?
007242  001002                           BNE     2$                 ;BR IF NOT
007244  004737  013464                   JSR     PC,CMPAR           ;NO ERROR, COMPARE DATA
007250  000207                  2$:      RTS     PC                 ;RETURN

                                ;ORDER TERMINATED WITH AN ERROR - PROCESS THE ERROR
```

```

007252  032760  000200  000016  ERPROC: BIT     #BIT07,$STATUS(R0) ;DONE BIT SET ?
007260  001402                          BEQ     ERPRC1             ;BR IF ORDER DIDN'T COMPLETE NORMALLY
007262  000137  007646                  JMP     DONE               ;PROCESS ERROR WITH 'DONE' BIT SET

                                ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE NOT' BITS

007266  032760  010000  000016  ERPRC1: BIT     #BIT12,$STATUS(R0) ;SEE IF DRIVE WAS UNSAFE
007274  001025                          BNE     PUNSAF             ;BR IF YES
007276  032760  004000  000016          BIT     #BIT11,$STATUS(R0) ;PARITY ERROR OCCURRED
007304  001043                          BNE     UCPAR              ;BR IF IT DID
007306  032760  002000  000016          BIT     #BIT10,$STATUS(R0) ;FATAL PARITY ERROR?
007314  001046                          BNE     FALPAR             ;BR IF THERE IS ONE
007316  032760  001000  000016          BIT     #BIT09,$STATUS(R0) ;TIMEOUT?
007324  001070                          BNE     SWTIM              ;BR IF YES
007326  032760  040002  000016          BIT     #BIT14!BIT01,$STATUS(R0)  ;DRIVE WENT OFFLINE ?
007334  001103                          BNE     OFLIN              ;BR IF IT DID
007336  032760  000004  000016          BIT     #BIT2,$STATUS(R0)  ;PORT REQUEST TIME OUT ?
007344  001121                          BNE     PRTIM              ;BR IF IT DID
007346  000207                          RTS     PC                 ;ERROR. RETURN

                                ;DRIVE IS PERSISTENTLY UNSAFE

007350  104401  001203          PUNSAF: TYPE    ,$CRLF             ;CR-LF
007354  004737  021112                  JSR     PC,LINE1           ;PRINT LINE 1 OF ERROR MESSAGE
007360  104414  066562                  DISPLY  ,EM12              ;PERSISTENT DEVICE UNSAFE MESSAGE
007364  004737  021166                  JSR     PC,LINE2           ;PRINT LINE 2 OF ERROR MESSAGE
007370  004737  021574                  JSR     PC,LINE3           ;PRINT LINE 3 OF ERROR MESSAGE
007374  004737  022244                  JSR     PC,LINE4           ;PRINT LINE 4 OF THE ERROR MESSAGE
007400  004737  024340                  JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
007404  004737  022654                  JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
007410  000137  030056                  JMP     DROP               ;DROP THE DRIVE

                                ;UNCORRECTABLE MASSBUS PARITY ERROR OCCURRED

007414  104401  001203          UCPAR:  TYPE    ,$CRLF             ;CR-LF
007420  104401  001203                  TYPE    ,$CRLF             ;CR-LF
007424  104401  066464                  TYPE    ,EM10              ;'UNCORRECTABLE PARITY ERROR' MESSAGE
007430  000406                          BR      FALPR1             ;FINISH PROCESSING THE ERROR

                                ;'FATAL' MASSBUS PARITY ERROR OCCURRED

007432  104401  001203          FALPAR: TYPE    ,$CRLF             ;CR-LF
007436  104401  001203                  TYPE    ,$CRLF             ;CR-LF
007442  104401  066527                  TYPE    ,EM11              ;'FATAL PARITY ERROR' MESSAGE

007446  104401  074064          FALPR1: TYPE    ,MSGON             ;TYPE 'ON'
007452  104401  073331                  TYPE    ,UNTMSG            ;TYPE 'DRIVE'
007456  013746  001346                  MOV     UNIT,-(SP)         ;;SAVE UNIT FOR TYPEOUT
                                                                   ;;TYPE DRIVE NUMBER
007462  104403                          TYPOS                      ;;GO TYPE--OCTAL ASCII
007464     002                          .BYTE   2                  ;;TYPE 2 DIGIT(S)
007465     000                          .BYTE   0                  ;;SUPPRESS LEADING ZEROS
007466  004737  024340                  JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
007472  032777  100000  171454          BIT     #SW15,@SWR         ;HALT ON ERROR ?
007500  001401                          BEQ     1$                 ;BR IF NOT
007502  000000                          HALT                       ;ERROR HALT
```

```
007504  000207                          1$:     RTS     PC

                                        ;SOFTWARE TIMEOUT OCCURRED

007506  004737  021112                  SWTIM:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007512  104414  066613                          DISPLY  ,EM13           ;PRINT THE TIME OUT MESSAGE
007516  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007522  004737  021574                          JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
007526  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
007532  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007536  004737  022654                          JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007542  000207                                  RTS     PC              ;RETURN

                                        ;DRIVE WENT OFFLINE

007544  104401  001203                  OFLIN:  TYPE    ,$CRLF          ;CR-LF
007550  004737  021112                          JSR     PC,LINE1        ;PRINT LINE 1 OF THE ERROR MESSAGE
007554  104414  066665                          DISPLY  ,EM14           ;PRINT OFFLINE MESSAGE
007560  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF THE ERROR MESSAGE
007564  004737  021574                          JSR     PC,LINE3        ;PRINT LINE 3 OF THE ERROR MESSAGE
007570  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF THE ERROR MESSAGE
007574  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007600  004737  022654                          JSR     PC,LINE7        ;PRINT LINE 7 OF THE ERROR MESSAGE
007604  000137  030056                          JMP     DROP            ;DROP THE DRIVE

                                        ;PORT REQUEST TIMEOUT ERROR

007610  004737  021112                  PRTIM:  JSR     PC,LINE1        ;TYPE LINE 1 OF THE ERROR MESSAGE
007614  104414  066710                          DISPLY  ,EM15           ;PRINT PORT TIME OUT MESSAGE
007620  004737  021166                          JSR     PC,LINE2        ;TYPE LINE 2 OF THE ERROR MESSAGE
007624  004737  021574                          JSR     PC,LINE3        ;TYPE LINE 3 OF THE ERROR MESSAGE
007630  004737  022244                          JSR     PC,LINE4        ;TYPE LINE 4 OF THE ERROR MESSAGE
007634  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007640  004737  022654                          JSR     PC,LINE7        ;TYPE LINE 7 OF THE ERROR MESSAGE
007644  000207                                  RTS     PC              ;RETURN

                                        ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE' BITS SET

007646  032760  000030  000016          DONE:   BIT     #BIT04!BIT03,$STATUS(R0) ;UNSAFE OCCURRED
007654  001402                                  BEQ     .+6             ;BR IF NOT
007656  000137  012754                          JMP     UNSAF           ;REPORT UNSAFE
007662  032760  040000  002126                  BIT     #BIT14,$RMCS2(R0)  ;IS 'WCE' SET ?
007670  001402                                  BEQ     .+6             ;BRANCH IF NOT SET
007672  000137  010630                          JMP     WCKER           ;WRITE CHECK ERROR
007676  032760  040000  002130                  BIT     #BIT14,$RMDS(R0) ;CHECK 'ERR'
007704  001002                                  BNE     1$              ;BR IF SET
007706  000137  012514                          JMP     TRFER           ;PROCESS 'TRE'
007712  032760  000400  002132          1$:     BIT     #BIT08,$RMER1(R0) ;'HCRC' SET?
007720  001402                                  BEQ     .+6             ;BR IF NOT
007722  000137  011156                          JMP     HCRCER          ;PROCESS 'HCRC'
007726  032760  000020  002132                  BIT     #BIT04,$RMER1(R0) ;'FMT' SET?
007734  001402                                  BEQ     .+6             ;BR IF NOT SET
007736  000137  011354                          JMP     CKFMT           ;CHECK FORMAT ERROR
007742  032760  000200  002132                  BIT     #BIT07,$RMER1(R0) ;'HCE' SET?
007750  001402                                  BEQ     .+6             ;BR IF NOT SET
007752  000137  011550                          JMP     CKHCE           ;CHECK 'HCE' ERROR
007756  032760  020000  002132                  BIT     #BIT13,$RMER1(R0) ;'OPI' SET?
```

```
    335 007764  001402                          BEQ     .+6                 ;BR IF NOT SET
    336 007766  000137  012050                  JMP     OPIER               ;REPORT 'OPI'
    337 007772  032760  000010  002132          BIT     #BIT3,$RMER1(R0)    ;'PAR' SET?
    338 010000  001402                          BEQ     .+6                 ;BR IF NOT SET
    339 010002  000137  012202                  JMP     PARER               ;REPORT 'PAR'
    340 010006  032760  000040  002132          BIT     #BIT5,$RMER1(R0)    ;'WCF' SET?
    341 010014  001402                          BEQ     .+6                 ;BR IF NOT SET
    342 010016  000137  012656                  JMP     WCFER               ;REPORT 'WCF'
    343 010022  032760  002000  002132          BIT     #BIT10,$RMER1(R0)   ;'IAE' SET?
    344 010030  001402                          BEQ     .+6                 ;BR IF NOT SET
    345 010032  000137  012274                  JMP     IAEER               ;REPORT 'IAE'
    346 010036  032760  004000  002132          BIT     #BIT11,$RMER1(R0)   ;'WLE' SET?
    347 010044  001402                          BEQ     .+6                 ;BR IF NOT SET
    348 010046  000137  012326                  JMP     WLEER               ;REPORT 'WLE'
    349 010052  032760  001000  002132          BIT     #BIT9,$RMER1(R0)    ;'AOE' SET?
    350 010060  001405                          BEQ     2$                  ;BR IF NOT SET
    351 010062  032760  002000  002130          BIT     #BIT10,$RMDS(R0)    ;'LST' SET?
    352 010070  001401                          BEQ     2$                  ;BR IF NOT SET
    353 010072  000207                          RTS     PC                  ;'AOE' & 'LST' SET, EXIT
    354 010074  032760  010000  002132  2$:     BIT     #BIT12,$RMER1(R0)    ;SEE IF 'DTE' SET
    355 010102  001402                          BEQ     .+6                 ;BR IF NOT
    356 010104  000137  012160                  JMP     DTEER               ;REPORT 'DTE' ERROR
    357 010110  005760  002132                  TST     $RMER1(R0)          ;SEE IF 'DCK' SET
    358 010114  100002                          BPL     .+6                 ;BR IF NOT
    359 010116  000137  010154                  JMP     DCKER               ;PROCESS 'DCK'
    360 010122  032760  060000  002160          BIT     #BIT14!BIT13,$RMER2(R0)  ;'SKI' OR 'OCYL' SET
    361 010130  001006                          BNE     3$                  ;BRANCH IF SKI, OCYL SET
    362 010132  032760  100000  002160          BIT     #BIT15,$RMER2(R0)   ;BAD SPOT ?
    363 010140  001004                          BNE     4$                  ;BRANCH IF SO (NO, OTHER ERROR)
    364 010142  000137  011322                  JMP     DRVER               ;REPORT ERROR
    365 010146  000137  012614          3$:     JMP     SKIER               ;REPORT SKI ERROR
    366 010152  000207                  4$:     RTS     PC                  ;EXIT FROM ERROR ANALYSIS  ROUT.
    367
    368                                 ;PROCESS DATA ('DCK') CHECK ERROR
    369
    370 010154  022760  010041  002162  DCKER:  CMP     #10041,$RMEC1(R0)   ;VALID POSITION COUNT ?
    371 010162  101407                          BLOS    1$                  ;BR IF NOT VALID
    372 010164  005760  002162                  TST     $RMEC1(R0)          ;POSITION COUNT 0 ?
    373 010170  001404                          BEQ     1$                  ;BR IF 0'S
    374 010172  005760  002164                  TST     $RMEC2(R0)          ;VALUE IN PATTERN REGISTER ?
    375 010176  001027                          BNE     4$                  ;BR IF YES
    376 010200  000431                          BR      6$                  ;DATA CHECK ERROR
    377 010202  004737  021112          1$:     JSR     PC,LINE1            ;TYPE FIRST LINE OF ERROR MESSAGE
    378 010206  104414  070327                  DISPLY  ,EM45               ;TYPE 'ECC LOGIC ERROR'
    379 010212  004737  021166                  JSR     PC,LINE2            ;TYPE LINE 2 OF ERROR MESSAGE
    380 010216  004737  024340                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
    381 010222  012737  000003  001352          MOV     #3,RETRY            ;RETRY COUNT
    382 010230  004737  016026                  JSR     PC,$RETRY           ;RETRY THE ORDER
    383 010234  000403                          BR      2$                  ;RETRY WAS NOT SUCCESSFUL
    384 010236  004737  022572                  JSR     PC,LINE6C           ;TYPE LINE 6C OF ERROR MESSAGE
    385 010242  000402                          BR      3$                  ;FINISH THE ERROR REPORT
    386 010244  004737  022600          2$:     JSR     PC,LINE6D           ;TYPE LINE 6D OF ERROR MESSAGE
    387 010250  004737  022654          3$:     JSR     PC,LINE7            ;TYPE LINE 7 OF ERROR MESSAGE
    388 010254  000402                          BR      5$                  ;EXIT
    389 010256  004737  020760          4$:     JSR     PC,SPOTCK           ;SEE IF ERROR AT A BAD SPOT ON THE PACK
    390 010262  000207                  5$:     RTS     PC                  ;IT IS, DON'T REPORT IT
    391 010264  004737  021112          6$:     JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
```

```
010270  104414  066765                    DISPLY  ,EM21              ;DATA CHECK ERROR
010274  004737  021166            DCKER1: JSR     PC,LINE2           ;PRINT LINE 2 OF ERROR MESSAGE
010300  004737  021574                    JSR     PC,LINE3           ;PRINT LINE 3 OF ERROR MESSAGE
010304  004737  022244                    JSR     PC,LINE4           ;PRINT LINE 4 OF ERROR MESSAGE
010310  004737  015450                    JSR     PC,PRTBAD          ;SEE IF BAD SECTOR TO BE PRINTED
010314  012737  110100  001350            MOV     #BIT15!BIT12!BIT06,MASK  ;LOAD ERROR MASK
010322  032760  010100  002132            BIT     #BIT12!BIT06,$RMER1(R0)  ;CHECK 'DTE' & 'ECH'
010330  001003                            BNE     1$                 ;BR IF SET
010332  004737  022544                    JSR     PC,LINE6           ;PRINT LINE 6 OF ERROR MESSAGE
010336  000477                            BR      8$                 ;FINISH THE ERROR REPORT
010340  012737  000020  001352    1$:     MOV     #16.,RETRY         ;RETRY COUNT
010346  005001                            CLR     R1                 ;R1 IS OFFSET CODE POINTER


010350  004737  017024            2$:     JSR     PC,GODRIV          ;RETRY
010354  005760  000016            3$:     TST     $STATUS(R0)        ;TEST FOR DONE
010360  001775                            BEQ     3$                 ;BR IF NOT DONE
010362  100075                            BPL     10$                ;BR IF NOT ERROR
010364  032760  000200  000016            BIT     #BIT7,$STATUS(R0)    ;SEE IF ORDER TERMINIATED NORMALLY
010372  001006                            BNE     14$                ;BR IF NOT
010374  004737  024340                    JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
010400  104414  072455                    DISPLY  ,LIN8M             ;'DIFFERENT ERROR DURING RETRY'
010404  000137  007266                    JMP     ERPRC1             ;SEE WHICH ERROR
010410  033760  001350  002132    14$:    BIT     MASK,$RMER1(R0)    ;LOOK AT CURRENT ERROR
010416  001430                            BEQ     5$                 ;BR IF DIFFERENT ERROR
010420  032760  010100  002132            BIT     #BIT12!BIT6,$RMER1(R0)    ;'ECH' OR 'DTE' STILL SET ?
010426  001437                            BEQ     7$                 ;BR IF NEITHER SET
010430  105237  001353                    INCB    RETRY+1            ;INCREMENT RETRY COUNT
010434  123737  001352  001353            CMPB    RETRY,RETRY+1      ;DONE ?
010442  001342                            BNE     2$                 ;BR IF NOT
010444  005201                            INC     R1                 ;INCREMENT TABLE INDEX
010446  116137  002470  066071            MOVB    OFFCOD(R1),GENDPB+$FMT    ;OFFSET CODE
010454  001435                            BEQ     9$                 ;BR IF END OF OFFSET TABLE
010456  062737  000002  001352            ADD     #2,RETRY           ;NEW RETRY LIMIT
010464  004737  015710                    JSR     PC,OFFST           ;OFFSET
010470  005737  066106            4$:     TST     GENDPB+$STATUS     ;SEE IF FINISHED WITH OFFSET
010474  001775                            BEQ     4$                 ;BR IF NOT
010476  100324                            BPL     2$                 ;BR IF NO ERROR PERFORMING OFFSET
010500  004737  023112            5$:     JSR     PC,LINE8           ;PRINT LINE 8 OF ERROR MESSAGE
010504  004737  024244            6$:     JSR     PC,INCHRD          ;INCREMENT 'HARD' ERROR COUNT
010510  004737  024340                    JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
010514  004737  022654                    JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
010520  004737  015450                    JSR     PC,PRTBAD          ;PRINT THE BAD SECTOR
010524  000436                            BR      13$                ;CLEAN UP AND RETURN
010526  004737  022564            7$:     JSR     PC,LINE6B          ;PRINT LINE 6B OF ERROR MESSAGE
010532  004737  022502                    JSR     PC,LINE5B          ;PRINT LINE 5B OF THE ERROR MESSAGE
010536  004737  024220            8$:     JSR     PC,INCSOF          ;INCREMENT 'SOFT' ERROR COUNT
010542  004737  014710                    JSR     PC,ECC             ;CORRECT THE ERROR USING ECC AND CHECK IT
010546  000407                            BR      11$                ;COMPARE THE BUFFER
010550  004737  022600            9$:     JSR     PC,LINE6D          ;PRINT LINE 6D OF ERROR MESSAGE
010554  000753                            BR      6$                 ;INCREMENT ERROR COUNT
010556  004737  022556            10$:    JSR     PC,LINE6A          ;PRINT LINE 6A OF ERROR MESSAGE
010562  004737  024220                    JSR     PC,INCSOF          ;INCREMENT 'SOFT' ERROR COUNT
010566  012737  000001  001400    11$:    MOV     #1,FRSTER          ;SET PROCESSING 'DCKER' INDICATOR
010574  004737  013502                    JSR     PC,CMPARD          ;COMPARE THE BUFFER
010600  105737  001401                    TSTB    FRSTER+1           ;ERROR IN COMPARE ?
010604  100406                            BMI     13$                ;BRANCH IF ERROR
```

```
010606  004737  024340                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
010612  104414  072655                  DISPLY  ,LIN9G              ;'DATA COMPARE OK' MESSAGE
010616  004737  022654          12$:    JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
010622  004737  007040          13$:    JSR     PC,REFMT            ;REFORMAT THE ERROR SECTOR
010626  000207                          RTS     PC                  ;RETURN

                                ;WRITE CHECK ERROR PROCESSING

010630  032760  100000  002132  WCKER:  BIT     #BIT15,$RMER1(R0)   ;SEE IF 'DCK' SET ALSO
010636  001034                          BNE     2$                  ;BR IF IT IS
010640  004737  021112                  JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
010644  104414  067071                  DISPLY  ,EM23               ;PRINT WCE & DCK NOT
010650  005037  001350                  CLR     MASK                ;CLEAR ERROR MASK
010654  004737  021166                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
010660  004737  021574                  JSR     PC,LINE3            ;PRINT LINE 3 OF ERROR MESSAGE
010664  004737  022244                  JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
010670  004737  022334                  JSR     PC,LINE5            ;PRINT LINE 5 OF ERROR MESSAGE
010674  004737  024340                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
010700  012737  000003  001352          MOV     #3,RETRY            ;RETRY LIMIT
010706  004737  016026                  JSR     PC,$RETRY           ;RETRY THE OPERATION
010712  000403                          BR      1$                  ;RETRY UNSUCESSFUL
010714  004737  022572                  JSR     PC,LINE6C           ;PRINT LINE 6C OF ERROR MESSAGE
010720  000502                          BR      8$                  ;FINISH PROCESSING THE ERROR
010722  004737  022600          1$:     JSR     PC,LINE6D           ;PRINT LINE 6D OF ERROR MESSAGE
010726  000506                          BR      10$                 ;FINISH PROCESSING THE ERROR
010730  004737  020760          2$:     JSR     PC,SPOTCK           ;SEE IF ERROR AT BAD SPOT ON THE PACK
010734  000507                          BR      11$                 ;EXIT IF AT BAD SPOT ON PACK
010736  004737  021112                  JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
010742  012737  067016  010770          MOV     #EM22,13$           ;ASSUME THAT EM22 WILL BE PRINTED
010750  032760  040000  002126          BIT     #BIT14,$RMCS2(R0)   ;DID 'WCK' ALSO SET ?
010756  001003                          BNE     12$                 ;BR IF IT DID
010760  012737  067717  010770          MOV     #EM37,13$           ;MESSAGE FOR 'DCK' AND 'WCK' NOT DURING
                                                                    ;WRITE CHECK
010766  104414                  12$:    DISPLY                      ;TYPE THE ERROR MESSAGE
010770  000000                  13$:    .WORD   0                   ;MESSAGE ADDRESS GOES HERE
010772  004737  021166                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
010776  004737  021574                  JSR     PC,LINE3            ;PRINT LINE 3 OF ERROR MESSAGE
011002  004737  022244                  JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
011006  004737  022334                  JSR     PC,LINE5            ;PRINT LINE 5 OF ERROR MESSAGE
011012  032760  000100  002132          BIT     #BIT06,$RMER1(R0)   ;ECH SET ALSO ?
011020  001442                          BEQ     8$                  ;FINISH PROCESSING THE ERROR
011022  012737  000020  001352  3$:     MOV     #16.,RETRY          ;RETRY LIMIT - 16 (10)
011030  004737  017024          4$:     JSR     PC,GODRIV           ;RETRY THE ORDER
011034  005760  000016          5$:     TST     $TATUS(R0)          ;ORDER FINISHED ?
011040  001775                          BEQ     5$                  ;BR IF NOT
011042  100405                          BMI     6$                  ;BR IF ERROR ON ORDER
011044  105237  001353                  INCB    RETRY+1             ;INCREMENT RETRY COUNT
011050  004737  022572                  JSR     PC,LINE6C           ;PRINT LINE 6C OF ERROR MESSAGE
011054  000431                          BR      9$                  ;FINISH ERROR PROCESSING
011056  105237  001353          6$:     INCB    RETRY+1             ;INCREMENT RETRY COUNT
011062  123737  001352  001353          CMPB    RETRY,RETRY+1       ;DONE ?
011070  001714                          BEQ     1$                  ;BR IF AT RETRY LIMIT
011072  032760  100000  002132          BIT     #BIT15,$RMER1(R0)   ;'DCK' SET
011100  001407                          BEQ     7$                  ;BR IF NOT - DIFFERENT ERROR
011102  032760  000100  002132          BIT     #BIT06,$RMER1(R0)   ;'ECH' ALSO SET ?
011110  001347                          BNE     4$                  ;BR IF IT IS, RETRY ORDER
011112  004737  022572                  JSR     PC,LINE6C           ;PRINT LINE 6C OF ERROR MESSAGE
```

```
011116  000403                                BR      8$              ;FINISH PROCESSING ERROR
011120  004737  023112                7$:     JSR     PC,LINE8        ;PRINT LINE 8 - 'DIFFERENT ERROR '
011124  000405                                BR      9$              ;FINISH PROCESSING ERROR
011126  004737  024340                8$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011132  004737  022654                        JSR     PC,LINE7        ;FINISH THE ERROR MESSAGE
011136  000406                                BR      11$             ;EXIT
011140  004737  024340                9$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011144  004737  022654                10$:    JSR     PC,LINE7        ;FINISH THE ERROR MESSAGE
011150  004737  007040                        JSR     PC,REFMT        ;REFORMAT THE SECTOR IN ERROR
011154  000207                        11$:    RTS     PC              ;RETURN

                                      ;REPORT 'HCRC' ERROR

011156  004737  020760                HCRCER: JSR     PC,SPOTCK       ;SEE IF ERROR AT PACK BAD SPOT
011162  000456                                BR      3$              ;EXIT IF IT IS
011164  004737  023176                        JSR     PC,READDR       ;READ ERROR SECTOR HEADER
011170  004737  015734                        JSR     PC,READHD       ;GET THE HEAD INFORMATION
011174  004737  021112                        JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011200  104414  066744                        DISPLY  ,EM20           ;REPORT 'HCRC'
011204  004737  021166                        JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011210  004737  021574                        JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011214  004737  022244                        JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011220  004737  022434                        JSR     PC,LINE5A       ;PRINT THE HEADER INFORMATION
011224  032760  040000  002126                BIT     #BIT14,$RMCS2(R0)   ;'WCE' ERROR ALSO ?
011232  001402                                BEQ     1$              ;BR IF NOT
011234  004737  022334                        JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
011240  004737  024220                1$:     JSR     PC,INCSOF       ;INCREMENT 'SOFT' ERROR COUNT
011244  004737  024340                        JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011250  012737  000400  001350                MOV     #BIT8,MASK      ;SET ERROR MASK
011256  012737  000003  001352                MOV     #3,RETRY        ;RETRY LIMIT
011264  004737  016026                        JSR     PC,$RETRY       ;RETRY ORDER
011270  000405                                BR      2$              ;RETRY NOT SUCESSFUL
011272  004737  022572                        JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
011276  004737  022654                        JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011302  000406                                BR      3$              ;EXIT
011304  004737  022600                2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
011310  004737  022654                        JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011314  004737  007040                        JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
011320  000207                        3$:     RTS     PC              ;RETURN

                                      ;REPORT DRIVE ERROR

011322  004737  021112                DRVER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011326  104414  067361                        DISPLY  ,EM30           ;REPORT DRIVE ERROR
011332  004737  021166                        JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011336  004737  021574                        JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011342  004737  024340                        JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011346  004737  022654                        JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011352  000207                                RTS     PC              ;RETURN

                                      ;PROCESS FORMAT ('FER') ERROR

011354  032760  000400  002132        CKFMT:  BIT     #BIT8,$RMER1(R0) ;'HCRC' SET ON ORIGINAL ERROR ?
011362  001402                                BEQ     1$              ;BR IF NOT SET
011364  000137  011156                        JMP     HCRCER          ;REPORT HCRC ERROR

011370  004737  023176                1$:     JSR     PC,READDR       ;GET CORRECTED TRACK & SECTOR ADDRSSES
```

```
011374  004737  015734                  JSR     PC,READHD               ;READ HEADER
011400  032737  000400  066124          BIT     #BIT8,GENREG+RMER1 ;'HCRC' SET WHEN HEADER READ?
011406  001002                          BNE     2$                      ;BR IF 'HCRC' SET
011410  000137  012354                  JMP     FMTER                   ;NO, ERROR IS 'FMT' ONLY

011414  004737  020760          2$:     JSR     PC,SPOTCK               ;SEE IF ERROR AT BAD SPOT ON THE PACK
011420  000452                          BR      5$                      ;EXIT IF IT IS
011422  004737  021112                  JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
011426  104414  067150                  DISPLY  ,EM24                   ;HEADER READ ERROR - FMT BIT DROPPED UP
011432  004737  021166                  JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
011436  004737  021574                  JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
011442  004737  022244                  JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
011446  032760  040000  002126          BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
011454  001402                          BEQ     3$                      ;BR IF NOT
011456  004737  022334                  JSR     PC,LINE5                ;DISPLAY WORDS WHICH CAUSED 'WCE'
011462  004737  022434          3$:     JSR     PC,LINE5A               ;DISPLAY HEADER
011466  004737  024220                  JSR     PC,INCSOF               ;INCREMENT SOFT ERROR COUNT
011472  004737  024340                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
011476  012737  000020  001350          MOV     #BIT4,MASK              ;SET ERROR MASK
011504  012737  000003  001352          MOV     #3,RETRY                ;RETRY LIMIT
011512  004737  016026                  JSR     PC,$RETRY               ;RETRY THE ORDER
011516  000405                          BR      4$                      ;RETRY NOT SUCESSFUL
011520  004737  022572                  JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
011524  004737  022654                  JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
011530  000406                          BR      5$                      ;EXIT
011532  004737  022600          4$:     JSR     PC,LINE6D               ;PRINT LINE 6D OF ERROR MESSAGE
011536  004737  022654                  JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
011542  004737  007040                  JSR     PC,REFMT                ;REFORMAT THE ERROR SECTOR
011546  000207                  5$:     RTS     PC                      ;RETURN

                                ;PROCESS HEADER COMPARE ('HCE') ERROR

011550  032760  000400  002132  CKHCE:  BIT     #BIT8,$RMER1(R0) ;HCRC SET ON ORIGINAL ERROR ?
011556  001402                          BEQ     1$                      ;BR IF NOT SET
011560  000137  011156                  JM.     HCRCER                  ;REPORT HEADER CRC ERROR
011564  004737  023176          1$:     JSR     PC,READDR               ;GET CURRENT SECTOR & TRACK ADDRS
011570  004737  015734                  JSR     PC,READHD               ;READ HEADER OF CURRENT SECTOR
011574  032737  000400  066124          BIT     #BIT8,GENREG+RMER1 ;'HCRC' SET ?
011602  001016                          BNE     3$                      ;BR IF SET
011604  042737  150000  076746          BIC     #150000,CYLNDR  ;CLEAR FORMAT,MFG,USER BITS FROM HEADER
011612  026037  002152  076746          CMP     $RMDC(R0),CYLNDR  ;CORRECT CYLINDER ?
011620  001402                          BEQ     2$                      ;BR IF IT IS
011622  000137  011774                  JMP     POSER                   ;REPORT POSITIONING ERROR
011626  052737  150000  076746  2$:     BIS     #150000,CYLNDR  ;RESTORE THE FORMAT,MFG,USER BITS
011634  000137  012432                  JMP     HCEER                   ;REPORT 'HCE' ERROR

011640  004737  020760          3$:     JSR     PC,SPOTCK               ;SEE IF ERROR AT BAD SPOT
011644  000452                          BR      6$                      ;EXIT IF IT IS
011646  004737  021112                  JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
011652  104414  067216                  DISPLY  ,EM25                   ;HEADER READ ERROR - 'HCE' SET
011656  004737  021166                  JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
011662  004737  021574                  JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
011666  004737  022244                  JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
011672  032760  040000  002126          BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
011700  001402                          BEQ     4$                      ;BR IF NOT
011702  004737  022334                  JSR     PC,LINE5                ;DISPLAY WORDS WHICH CAUSED 'WCE'
011706  004737  022434          4$:     JSR     PC,LINE5A               ;PRINT LINE 5 OF ERROR MESSAGE
```

```
011712  004737  024220                    JSR     PC,INCSOF          ;INCREMENT SOFT ERROR COUNT
011716  004737  024340                    JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
011722  012737  000200  001350            MOV     #BIT7,MASK         ;SET ERROR MASK
011730  012737  000003  001352            MOV     #3,RETRY           ;RETRY LIMIT
011736  004737  016026                    JSR     PC,$RETRY          ;RETRY THE ORDER
011742  000405                            BR      5$                 ;RETRY NOT SUCESSFUL
011744  004737  022572                    JSR     PC,LINE6C          ;PRINT LINE 6C OF ERROR MESSAGE
011750  004737  022654                    JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
011754  000406                            BR      6$                 ;EXIT
011756  004737  022600          5$:       JSR     PC,LINE6D          ;PRINT LINE 6D OF ERROR MESSAGE
011762  004737  022654                    JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
011766  004737  007040                    JSR     PC,REFMT           ;REFORMAT THE ERROR SECTOR
011772  000207                  6$:       RTS     PC                 ;RETURN


                                ;REPORT POSSIBLE POSITIONING ERROR

011774  004737  021112          POSER:    JSR     PC,LINE1           ;PRINT LINE 1 OF ERROR MESSAGE
012000  104414  070525                    DISPLY  ,EM51              ;PROGRAM DETECTED POSITIONING ERROR
012004  004737  021166                    JSR     PC,LINE2           ;PRINT LINE 2 OF ERROR MESSAGE
012010  004737  021622                    JSR     PC,LINE3C          ;PRINT LINE 3C OF ERROR MESSAGE
012014  052737  150000  076746            BIS     #150000,CYLNDR     ;RESTORE THE FORMAT BIT
012022  004737  022434                    JSR     PC,LINE5A          ;PRINT LINE 5A OF THE ERROR MESSAGE
012026  004737  024314                    JSR     PC,INCMIS          ;INCREMENT MISPOSITIONING COUNT
012032  004737  024340                    JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
012036  004737  022776                    JSR     PC,LINE7A          ;PRINT LINE 7A OF ERROR MESSAGE
012042  004737  015646                    JSR     PC,RECALT          ;RECALIBRATE
012046  000207                            RTS     PC                 ;EXIT


                                ;REPORT 'OPI' ERROR

012050  004737  020760          OPIER:    JSR     PC,SPOTCK          ;SEE IF ERROR AT BAD SPOT
012054  000207                            RTS     PC                 ;RETURN IF IT IS
012056  004737  021112                    JSR     PC,LINE1           ;PRINT LINE 1 OF ERROR MESSAGE
012062  104414  067413                    DISPLY  ,EM31              ;'OPI' ERROR
012066  004737  021166                    JSR     PC,LINE2           ;PRINT LINE 2 OF ERROR MESSAGE
012072  004737  021574                    JSR     PC,LINE3           ;PRINT LINE 3 OF ERROR MESSAGE
012076  004737  022244                    JSR     PC,LINE4           ;PRINT LINE 4 OF ERROR MESSAGE
012102  004737  024340                    JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
012106  012737  020000  001350            MOV     #BIT13,MASK        ;ERROR MASK
012114  012737  000003  001352  OPIER1:   MOV     #3,RETRY           ;RETRY LIMIT
012122  004737  016026                    JSR     PC,$RETRY          ;RETRY THE ORDER
012126  000405                            BR      1$                 ;RETRY UNSUCESSFUL
012130  004737  022572                    JSR     PC,LINE6C          ;PRINT LINE 6C OF ERROR MESSAGE
012134  004737  022654                    JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
012140  000207                            RTS     PC                 ;EXIT
012142  004737  022600          1$:       JSR     PC,LINE6D          ;PRINT LINE 6D OF ERROR MESSAGE
012146  004737  022654                    JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
012152  004737  007040                    JSR     PC,REFMT           ;REFORMAT THE ERROR SECTOR
012156  000207                            RTS     PC                 ;RETURN


                                ;REPORT 'DTE' ERROR

012160  004737  020760          DTEER:    JSR     PC,SPOTCK          ;SEE IF ERROR AT BAD SPOT
012164  000207                            RTS     PC                 ;RETURN IF IT IS
012166  004737  021112                    JSR     PC,LINE1           ;PRINT LINE 1 OF ERROR MESSAGE
012172  104414  067456                    DISPLY  ,EM32              ;'DTE' ERROR
012176  000137  010274                    JMP     DCKER1             ;FINISH PROCESSING THE 'DTE' ERROR
```

```

                                        ;REPORT 'PAR' ERROR

012202  004737  021112                  PARER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012206  104414  067511                          DISPLY  ,EM33           ;REPORT 'PAR'
012212  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012216  004737  021700                          JSR     PC,LINE3E       ;PRINT LINE 3E OF ERROR MESSAGE
012222  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012226  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012232  012737  000010  001350                  MOV     #BIT03,MASK     ;ERROR MASK
012240  012737  000003  001352                  MOV     #3,RETRY        ;RETRY LIMIT
012246  004737  016026                          JSR     PC,$RETRY       ;RETRY ORDER
012252  000405                                  BR      2$              ;RETRY UNSUCESSFUL
012254  004737  022572                          JSR     PC,LINE6C       ;RETRY SUCESSFUL
012260  004737  022654                  1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012264  000207                                  RTS     PC              ;EXIT
012266  004737  022600                  2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012272  000772                                  BR      1$              ;FINISH ERROR MESSAGE

                                        ;REPORT 'IAE' ERROR

012274  004737  021112                  IAEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012300  104414  067630                          DISPLY  ,EM35           ;REPORT 'IAE'
012304  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012310  004737  021766                          JSR     PC,LINE3F       ;PRINT LINE 3F OF ERROR MESSAGE
012314  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012320  004737  022654                          JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012324  000207                                  RTS     PC              ;RETURN

                                        ;REPORT 'WLE' ERROR

012326  004737  021112                  WLEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012332  104414  067666                          DISPLY  ,EM36           ;REPORT 'WLE'
012336  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012342  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012346  004737  022654                          JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012352  000207                                  RTS     PC              ;RETURN


                                        ;REPORT FORMAT ERROR

012354  004737  021112                  FMTER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012360  104414  067277                          DISPLY  ,EM26           ;FORMAT ERROR
012364  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012370  004737  021574                          JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012374  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012400  032760  040000  002126                  BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
012406  001402                                  BEQ     1$              ;BR IF NOT
012410  004737  022334                          JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
012414  004737  022434                  1$:     JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
012420  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012424  004737  022654                          JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012430  000207                                  RTS     PC

                                        ;REPORT HEADER COMPARE ERROR

012432  004737  021112                  HCEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012436  104414  067324                          DISPLY  ,EM27           ;HEADER COMPARE ERROR
```

```
012442  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012446  004737  021574                          JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012452  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012456  032760  040000  002126                  BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
012464  001402                                  BEQ     1$              ;BR IF NOT
012466  004737  022334                          JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
012472  004737  022434                  1$:     JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
012476  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012502  004737  022654                          JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012506  004737  007040                          JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
012512  000207                                  RTS     PC              ;RETURN

                                        ;PROCESS CONTROL/INTERFACE TRANSFER ERROR

012514  004737  021112                  TRFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012520  104414  070001                          DISPLY  ,EM40           ;RH11/RH70 OR UNIBUS TRANSFER ERROR
012524  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012530  004737  021574                          JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012534  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012540  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012544  032760  121400  002126                  BIT     #BIT15!BIT13!BIT9!BIT8,$RMCS2(R0)   ;'DLT','UPE','MXF', 'MDPE' SET ?
012552  001415                                  BEQ     2$              ;BR IF NONE SET
012554  012737  000003  001352                  MOV     #3,RETRY        ;RETRY LIMIT
012562  005037  001350                          CLR     MASK            ;CLEAR ERROR MASK
012566  004737  016026                          JSR     PC,$RETRY       ;RETRY THE OPERATION
012572  000403                                  BR      1$              ;RETURN HERE IF RETRY UNSUCESSFUL
012574  004737  022572                          JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
012600  000402                                  BR      2$              ;FINISH THE ERROR REPORT
012602  004737  022600                  1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012606  004737  022654                  2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012612  000207                                  RTS     PC

                                        ;PROCESS 'SKI' OR 'OCYL' ERRORS

012614  004737  021112                  SKIER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012620  104414  070467                          DISPLY  ,EM50           ;'SKI' OR 'OCYL' ERROR
012624  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012630  004737  021610                          JSR     PC,LINE3B       ;PRINT LINE 3B OF ERROR MESSAGE
012634  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012640  004737  024270                          JSR     PC,INCSKI       ;INCREMENT 'SKI' OR 'OCYL' ERROR COUNT
012644  004737  022776                          JSR     PC,LINE7A       ;PRINT LINE 7A OF ERROR MESSAGE
012650  004737  015646                          JSR     PC,RECALT       ;RECALIBRATE
012654  000207                                  RTS     PC

                                        ;REPORT WRITE CLOCK FAILURE ('WCF')

012656  004737  021112                  WCFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012662  104414  067566                          DISPLY  ,EM34           ;REPORT WRITE CLOCK FAILURE
012666  004737  021166                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012672  004737  021602                          JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
012676  004737  022244                          JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012702  004737  024340                          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012706  004737  015450                          JSR     PC,PRTBAD       ;SEE IF BAD SECTOR TO BE PRINTED
012712  012737  000003  001352                  MOV     #3,RETRY        ;RETRY COUNT
012720  012737  000040  001350                  MOV     #BIT05,MASK     ;ERROR MASK
012726  004737  016026                          JSR     PC,$RETRY       ;RETRY THE ORDER
012732  000405                                  BR      2$              ;RETURN HERE IF RETRY UNSUCESSFUL
```

```
012734  004737  022572                   JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
012740  004737  022654           1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012744  000207                           RTS     PC
012746  004737  022600           2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012752  000772                           BR      1$


                                 ;PROCESS DRIVE UNSAFE ERROR

012754  004737  021112           UNSAF:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012760  104414  070570                   DISPLY  ,EM60           ;REPORT DRIVE UNSAFE
012764  004737  021166                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012770  004737  021574                   JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012774  004737  024340                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
013000  012737  000003  001352           MOV     #3,RETRY        ;RETRY COUNT
013006  004737  016026                   JSR     PC,$RETRY       ;RETRY THE ORDER
013012  000403                           BR      1$              ;RETRY WAS UNSUCESSFUL
013014  004737  022572                   JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
013020  000402                           BR      2$              ;CONTINUE WITH ERROR REPORT
013022  004737  022600           1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
013026  004737  022654           2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
013032  000207                           RTS     PC              ;RETURN


                                 ;REPORT AN 'UNKNOWN' DATA PATTERN

013034  105737  001400           NOMTCH: TSTB    FRSTER          ;FIRST ERROR IN THE SECTOR ?
013040  001013                           BNE     1$              ;BR IF NOT OR IF PROCESSING 'DCKER'
013042  004737  021112                   JSR     PC,LINE1        ;TYPE LINE 1 OF ERROR MESSAGE
013046  104414  070171                   DISPLY  ,EM43           ;'CAN'T MATCH DATA WITH PATTERN'
013052  004737  021166                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
013056  004737  021602                   JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
013062  004737  022244                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
013066  000404                           BR      2$              ;CONTINUE PROCESSING ERROR
013070  104414  070171           1$:     DISPLY  ,EM43           ;'CAN'T MATCH DATA WITH PATTERN'
013074  104414  001203                   DISPLY  ,$CRLF          ;CR-LF
013100  104414  072605           2$:     DISPLY  ,LIN9I          ;HEADER FOR DATA PRINTOUT
013104  010146                           MOV     R1,-(SP)        ;ADDRESS OF WORD 1
013106  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 1
013112  104414  073270                   DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
013116  012146                           MOV     (R1)+,-(SP)     ;ADDRESS OF WORD 1
013120  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 1
013124  104414  001203                   DISPLY  ,$CRLF          ;CR-LF
013130  010146                           MOV     R1,-(SP)        ;ADDRESS OF WORD 2
013132  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 2
013136  104414  073270                   DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
013142  012146                           MOV     (R1)+,-(SP)     ;ADDRESS OF WORD 2
013144  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 2
013150  104414  001203                   DISPLY  ,$CRLF          ;CR-LF
013154  010146                           MOV     R1,-(SP)        ;ADDRESS OF WORD 3
013156  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 3
013162  104414  073270                   DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
013166  012146                           MOV     (R1)+,-(SP)     ;ADDRESS OF WORD 3
013170  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 3
013174  104414  001203                   DISPLY  ,$CRLF          ;CR-LF
013200  010146                           MOV     R1,-(SP)        ;ADDRESS OF WORD 4
013202  004737  023124                   JSR     PC,LINOCT       ;TYPE WORD 4
013206  104414  073270                   DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
013212  012146                           MOV     (R1)+,-(SP)     ;ADDRESS OF WORD 4
```

```
        013214  004737  023124                  JSR     PC,LINOCT       ;TYPE WORD 4
        013220  104414  001203                  DISPLY  ,$CRLF          ;CR-LF
    832 013224  062701  000770                  ADD     #<252.*2.>,R1   ;INCREMENT BUFFER POINTER
    833 013230  162737  000400  001412          SUB     #256.,CMCNT     ;ADJUST WORD COUNT FOR MATCH
    834 013236  132760  000001  000024          BITB    #BIT00,$CODE(R0) ;HEADER OPERATION INVOLVED ?
    835 013244  001405                          BEQ     3$              ;NO
    836 013246  062701  000004                  ADD     #4,R1           ;ADJUST BUFFER ADDRESS
    837 013252  162737  000002  001412          SUB     #2,CMCNT        ;ADJUST WORD COUNT
    838 013260  005002                  3$:     CLR     R2              ;CLEAR 'WORDS TO COMPARE' COUNT IN R2
    839 013262  112737  000001  001400          MOVB    #1,FRSTER       ;SET 'NOT FIRST ERROR' INDICATOR
    840 013270  112737  177777  001401          MOVB    #-1,FRSTER+1    ;SET ERROR FOUND INDICATOR
    841 013276  013737  001476  001410          MOV     CMPLMT,LIMIT    ;RESET THE COMPARE ERROR TYPEOUT LIMIT
    842 013304  000207                          RTS     PC              ;RETURN
    843
    844                                 ;CHECK ERROR BITS IN THE RH/RM REGISTERS
    845
    846 013306  032760  060000  002116  CKERR:  BIT     #60000,$RMCS1(R0)  ;SEE IF 'TRE' OR 'MCPE' SET
    847 013314  001012                          BNE     1$              ;BR IT EITHER SET
    848 013316  032760  177400  002126          BIT     #177400,$RMCS2(R0)  ;SEE IF ERROR BITS IN CS2 SET
    849 013324  001006                          BNE     1$              ;BR IF ANY SET
    850 013326  005760  002132                  TST     $RMER1(R0)      ;ANY BITS SET IN ER1
    851 013332  001003                          BNE     1$              ;BR IF ANY SET
    852 013334  005760  002160                  TST     $RMER2(R0)      ;ANY BITS SET IN ER2 ?
    853 013340  001416                          BEQ     2$              ;BR IF NONE SET
    854 013342  004737  021112          1$:     JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
    855 013346  104414  070236                  DISPLY  ,EM44           ;ERROR BITS SET, BUT 'SC' OR 'TRE' NOT SET
    856 013352  004737  021166                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
    857 013356  004737  021574                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
    858 013362  004737  022244                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
    859 013366  004737  024340                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
    860 013372  004737  022654                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
    861 013376  000207                  2$:     RTS     PC              ;RETURN
    862
    863                                 ;CHECK BUS ADDRESS REGISTER & WORD COUNT REGISTER
    864
    865 013400  005760  002120          CKBUS:  TST     $RMWC(R0)       ;CHECK WORD COUNT
    866 013404  001010                          BNE     1$              ;BR IF NOT ZERO
    867 013406  016046  000020                  MOV     $WRDL(R0),-(SP) ;WORD LENGTH
    868 013412  006316                          ASL     (SP)            ;CHANGE INTO BYTE COUNT
    869 013414  066016  000006                  ADD     $BUF(R0),(SP)   ;ADD THE STARTING LOCATION
    870 013420  022660  002122                  CMP     (SP)+,$RMBA(R0) ;BUFFER ADDRESS PROPER ?
    871 013424  001416                          BEQ     2$              ;BR IF OK
    872 013426  004737  021112          1$:     JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
    873 013432  104414  070044                  DISPLY  ,EM41           ;BUS ADDRESS OR WORD COUNT INCORRECT
    874 013436  004737  021166                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
    875 013442  004737  021632                  JSR     PC,LINE3D       ;PRINT LINE 3D OF ERROR MESSAGE
    876 013446  004737  022244                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
    877 013452  004737  024340                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
    878 013456  004737  022654                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
    879 013462  000207                  2$:     RTS     PC
    880
    881                                 ;COMPARE THE BUFFER
    882
    883 013464  132760  000004  000024  CMPAR:  BITB    #BIT02,$CODE(R0)  ;SEE IF READ ORDER
    884 013472  001001                          BNE     1$              ;BR IF IT IS
    885 013474  000207                          RTS     PC              ;RETURN
    886 013476  005037  001400          1$:     CLR     FRSTER          ;CLEAR 'FIRST ERROR' INDICATOR
```

```
013502 032777 000002 165444 CMPARD: BIT     #SW01,@SWR          ;IS SWITCH 1 SET?
013510 001401                       BEQ     1$                  ;BR IF NOT
013512 000207                       RTS     PC                  ;YES, DON'T COMPARE
013514 005037 001406        1$:     CLR     ERCTR               ;CLEAR THE ERROR COUNTER
013520 016001 000006                MOV     $BUF(R0),R1         ;BUFFER ADDRESS
013524 016037 000020 001412         MOV     $WRDL(R0),CMCNT      ;WORD COUNT TO WORKING LOCATION
013532 066037 002120 001412         ADD     $RMWC(R0),CMCNT      ;CALCULATE ACTUAL WORDS TRANSFERED
013540 016037 000012 001414         MOV     $CYL(R0),CMCYL      ;CYLINDER ADDRESS WORKING LOCATION
013546 052737 010000 001414         BIS     #BIT12,CMCYL        ;SET FORMAT BIT
013554 052737 140000 001414         BIS     #140000,CMCYL       ;SET MFG AND USER BITS
013562 016037 000010 001416         MOV     $SEC(R0),CMSEC      ;SECTOR & TRACK ADDRESSES TO WORKING LOCNS
013570 013737 001476 001410         MOV     CMPLMT,LIMIT        ;DISPLAY LIMIT
013576 005237 001410                INC     LIMIT               ;CONVERT PARAMETER INTO LIMIT VALUE
013602 012737 177777 001376 CMSTR:  MOV     #-1,ZROIND          ;CLEAR THE 'ZERO'S' INDICATOR
013610 005037 001402                CLR     SAVER1              ;CLEAR THE R1 SAVE WORD
013614 005037 001404                CLR     SAVER5              ;CLEAR THE R5 SAVE WORD
013620 023760 001412 000022         CMP     CMCNT,$SSEC(R0)      ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
013626 101005                       BHI     1$                  ;BR IF IT IS
013630 013702 001412                MOV     CMCNT,R2            ;LESS THAN, USE REMAINING BUFFER
013634 005037 001412                CLR     CMCNT               ;SET COUNTER TO 0
013640 000405                       BR      2$                  ;
013642 016002 000022        1$:     MOV     $SSEC(R0),R2        ;COMPARE SECTOR
013646 166037 000022 001412         SUB     $SSEC(R0),CMCNT     ;DECREMENT WORD COUNT
013654 126027 000024 000005 2$:     CMPB    $CODE(R0),#5        ;READ HEADER & DATA?
013662 001025                       BNE     CMDAT               ;BR IF NOT
013664 052711 140000        CMHED:  BIS     #BIT15!BIT14,(R1)    ;SET BIT14,BIT15 IN CASE BAD SPOT ENCOUNTER
013670 023721 001414                CMP     CMCYL,(R1)+         ;CHECK CYLINDER
013674 001402                       BEQ     1$                  ;BR IF COMPARE OK
013676 004737 013724                JSR     PC,CMSTR2                   ;REPORT ERROR
013702 023721 001416        1$:     CMP     CMSEC,(R1)+         ;COMPARE SECTOR & TRACK
013706 001402                       BEQ     4$                  ;BR IF EQ
013710 004737 013724                JSR     PC,CMSTR2                   ;REPORT ERROR
013714 162702 000002        4$:     SUB     #2,R2               ;SUBTRACT HEADER LENGTH FROM SIZE
013720 003570                       BLE     CMPRX               ;BR IF FINISHED
013722 000405                       BR      CMDAT               ;COMPARE THE DATA PORTION
013724 005237 001406        CMSTR2: INC     ERCTR               ;INCREMENT THE ERROR COUNT
013730 004737 014310                JSR     PC,CMPRT            ;REPORT THE COMPARISON ERROR
013734 000207                       RTS     PC                  ;CHECK THE REST OF THE HEADER

013736 004737 014632        CMDAT:  JSR     PC,MATCH            ;FIND THE PATTERN
013742 000403                       BR      1$                  ;FOUND A PATTERN
013744 004737 013034                JSR     PC,NOMTCH           ;RETURN HERE IF NO MATCH WITH PATTERN MADE
013750 000456                       BR      7$                  ;BYPASS COMPARE ROUTINE
013752 011405               1$:     MOV     (R4),R5             ;ADDRESS OF PATTERN ADDRESS IN R4
013754 012703 000020                MOV     #20,R3              ;R3 IS PATTERN POS COUNTER
013760 022125               2$:     CMP     (R1)+,(R5)+         ;COMPARE BUFFER WITH PATTERN
013762 001016                       BNE     4$                  ;BR IF NOT EQUAL
013764 005737 001406                TST     ERCTR               ;ERRORS DETECTED ?
013770 001406                       BEQ     3$                  ;BR IF NO ERRORS
013772 032777 000010 165154         BIT     #SW3,@SWR           ;SWITCH 3 SET ?
014000 001402                       BEQ     3$                  ;BR IF NOT SET
014002 004737 014310                JSR     PC,CMPRT            ;DISPLAY THE WORD
014006 005302               3$:     DEC     R2                  ;DECREMENT SIZE COUNT
014010 003436                       BLE     7$                  ;BR WHEN AT END
014012 005303                       DEC     R3                  ;DECREMENT PATT POS COUNT
014014 001361                       BNE     2$                  ;BR IF NOT AT END OF PATT
014016 000755                       BR      1$                  ;RESTART THE PATTERN
```

```
014020  005761  177776                  4$:     TST     -2(R1)                  ;IS MISCOMPARED CHARACTER=0
014024  001410                                  BEQ     5$                      ;BR IF YES
014026  012737  177777  001376                  MOV     #-1,ZROIND              ;SET NON-ZERO MISCOMPARED INDCATOR
014034  005237  001406                          INC     ERCTR                   ;INCREMENT THE ERROR COUNTER
014040  004737  014310                          JSR     PC,CMPRT                ;REPORT ERROR
014044  000760                                  BR      3$                      ;CONTINUE COMPARE
014046  105737  001400                  5$:     TSTB    FRSTER                  ;FIRST ERROR?
014052  100407                                  BMI     6$                      ;BR IF NOT
014054  005037  001376                          CLR     ZROIND                  ;SET THE ZERO INDICATOR
014060  010137  001402                          MOV     R1,SAVER1               ;SAVE CURRENT R1
014064  010537  001404                          MOV     R5,SAVER5               ;SAVE CURRENT R5
014070  000746                                  BR      3$                      ;CONTINUE COMPARE
014072  005737  001376                  6$:     TST     ZROIND                  ;ANY MISCOMPARIONS NOT ZEROS ?
014076  001743                                  BEQ     3$                      ;BR IF NONE-ALL ERRORS=ZERO
014100  004737  014310                          JSR     PC,CMPRT                ;REPORT ERROR
014104  000740                                  BR      3$                      ;CONTINUE COMPARING
014106  023727  001412  000004          7$:     CMP     CMCNT,#4                ;LAST 3WORDS ?
014114  002472                                  BLT     CMPRX                   ;YES
014116  126027  000024  000005                  CMPB    $CODE(R0),#5            ;READ HEAD AND DATA ?
014124  001414                                  BEQ     9$                      ;YES
014126  013702  001412                  8$:     MOV     CMCNT,R2                ;SET COUNTER = REMAIN BUFFER LENGTH
014132  020227  000004                          CMP     R2,#4                   ;LAST  3 WORDS ?
014136  002461                                  BLT     CMPRX                   ;YES,EXIT
014140  162737  000400  001412                  SUB     #256.,CMCNT             ;GREATER THAN A SECTOR ?
014146  003673                                  BLE     CMDAT                   ;NO,RETURN TO COMPARE LOOP
014150  012702  000400                          MOV     #256.,R2                ;SET COUNTER =SECTOR SIZE
014154  000670                                  BR      CMDAT                   ;RETURN TO COMPARE LOOP
014156  105237  001416                  9$:     INCB    CMSEC                   ;INCREMENT COUNTER
014162  123727  001416  000040                  CMPB    CMSEC,#32.              ;MAX SECTOR # ?
014170  103423                                  BLO     11$                     ;NO
014172  105037  001416                          CLRB    CMSEC                   ;RESET SECTOR #
014176  105237  001417                          INCB    CMTRK                   ;INCREMENT TRACK #
014202  004737  026546                          JSR     PC,GETLMT               ;GET ADDRESS LIMITS
014206  123737  001417  001450                  CMPB    CMTRK,TRKLMT            ;MAX TRACK # ?
014214  101411                                  BLOS    11$                     ;NO
014216  105037  001417                          CLRB    CMTRK                   ;RESET TRACK #
014222  005237  001414                          INC     CMCYL                   ;INCREMENT CYLINDER NUMBER
014226  023727  001414  151466                  CMP     CMCYL,#150000+822.      ;LAST CYLINDER ?
014234  103401                                  BLO     11$                     ;NO
014236  000421                                  BR      CMPRX                   ;NORMAL RETURN,NOT WRAP AROUND
014240  052711  140000                  11$:    BIS     #BIT15!BIT14,(R1)       ;SET BIT15/14 INCASE BAD SPOT ENCOUNTER
014244  023721  001414                          CMP     CMCYL,(R1)+             ;COMPARE 1ST HEADER WORD
014250  001402                                  BEQ     12$                     ;MATCH
014252  004737  013724                          JSR     PC,CMSTR2               ;NOT MATCH
014256  023721  001416                  12$:    CMP     CMSEC,(R1)+             ;SECOND WORD OF HEADER
014262  001402                                  BEQ     13$                     ;MATCH
014264  004737  013724                          JSR     PC,CMSTR2               ;NOT MATCH
014270  162737  000002  001412          13$:    SUB     #2,CMCNT                ;ADJUST WORD COUNT
014276  003401                                  BLE     CMPRX                   ;COMPARE IS DONE
014300  000712                                  BR      8$                      ;RETURN TO COMPARE LOOP

014302  004737  014564                  CMPRX:  JSR     PC,ENDCMP               ;PRINT LAST LINE IF ERRORS
014306  000207                                  RTS     PC

                                        ;TYPE DATA COMPARE ERRORS

014310  005737  001402                  CMPRT:  TST     SAVER1                  ;PRINT SAVED VALUES ?
```

```
014314  001010                          BNE     2$              ;BR IF NOT
014316  105737  001400                  TSTB    FRSTER          ;FIRST ERROR?
014322  100402                          BMI     1$              ;BR IF NOT
014324  004737  014404                  JSR     PC,4$           ;PRINT INITIAL MESSAGE INFO
014330  004737  014466          1$:     JSR     PC,8$           ;PRINT REMAINDER OF MESSAGE
014334  000422                          BR      3$              ;EXIT
014336                          2$:
014336  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
014340  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
014342  013701  001402                  MOV     SAVER1,R1       ;DISPLAY SAVED R1
014346  013705  001404                  MOV     SAVER5,R5       ;DISPLAY SAVED R5
014352  004737  014404                  JSR     PC,4$           ;PRINT INITIAL MESSAGE INFO
014356  004737  014466                  JSR     PC,8$           ;PRINT SAVED VALUES
014362  005037  001402                  CLR     SAVER1          ;CLEAR SAVED REGISTER INDICATORS
014366  005037  001404                  CLR     SAVER5          ;CLEAR THE OTHER ONE
014372  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
014374  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
014376  004737  014466                  JSR     PC,8$           ;PRINT REMAINDER OF MESSAGE
014402  000207                  3$:     RTS     PC              ;RETURN
014404  105737  001400          4$:     TSTB    FRSTER          ;FIRST ERROR ?
014410  100425                          BMI     7$              ;BR IF NOT
014412  001013                          BNE     5$              ;BR IF FIRST ERROR AND PROCESSING 'DCK' ERROR
014414  004737  021112                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
014420  104414  070110                  DISPLY  ,EM42           ;DATA COMPARE ERROR
014424  004737  021166                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
014430  004737  021602                  JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
014434  004737  022244                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
014440  000404                          BR      6$              ;GO TO TYPE HEADER
014442  104414  072500          5$:     DISPLY  ,LIN9B          ;HEADER MESSAGE OF PROCESSING 'DCK' ERROR
014446  104414  001203                  DISPLY  ,$CRLF          ;CR-LF
014452  104414  072527          6$:     DISPLY  ,LIN9H          ;DISPLAY HEADER
014456  012737  177777  001400          MOV     #-1,FRSTER      ;SET ERROR FLAG
014464  000207                  7$:     RTS     PC              ;RETURN
014466  005737  001410          8$:     TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
014472  001403                          BEQ     9$              ;BR IF IT HAS
014474  005337  001410                  DEC     LIMIT           ;DECREMENT LIMIT COUNTER
014500  001005                          BNE     10$             ;BR IF NOT AT LIMIT
014502  032777  000200  164444  9$:     BIT     #SW07,@SWR      ;PRINT ALL DATA COMPARE ERRORS ?
014510  001001                          BNE     10$             ;BR IF YES
014512  000207                          RTS     PC              ;RETURN
014514  010146                  10$:    MOV     R1,-(SP)        ;BUFFER ADDRESS
014516  162716  000002                  SUB     #2,(SP)         ;ADJUST ADDRESS
014522  004737  023124                  JSR     PC,LINOCT       ;TYPE IT
014526  104414  073270                  DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
014532  016546  177776                  MOV     -2(R5),-(SP)    ;PUT GOOD DATA ON THE STACK
014536  004737  023124                  JSR     PC,LINOCT       ;TYPE IT
014542  104414  073270                  DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
014546  016146  177776                  MOV     -2(R1),-(SP)    ;BAD DATA
014552  004737  023124                  JSR     PC,LINOCT       ;TYPE IT
014556  104414  001203                  DISPLY  ,$CRLF          ;CR-LF
014562  000207                          RTS     PC              ;RETURN

                                ;LAST LINE OF COMPARE ERROR REPORTING

014564  105737  001401          ENDCMP: TSTB    FRSTER+1        ;ANY COMPARE ERRORS FOUND ?
014570  001417                          BEQ     2$              ;BR IF NOT
014572  005737  001406                  TST     ERCTR           ;SEE HOW MANY ERRORS
```

```
014576  001410                                  BEQ     1$                  ;BR IF ONLY CAN'T MATCH PATTERN
014600  104414  072625                          DISPLY  ,LIN9E              ;'NUMBER OF ERRORS='
014604  013746  001406                          MOV     ERCTR,-(SP)         ;NUMBER OF ERRORS
014610  004737  023156                          JSR     PC,LINDEC           ;TYPE IT
014614  104414  001203                          DISPLY  ,$CRLF              ;CR-LF
014620  004737  024340                  1$:     JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
014624  004737  022654                          JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
014630  000207                          2$:     RTS     PC                  ;RETURN


                                        ;ROUTINE TO MATCH THE DATA WITH A PATTERN
                                        ;CALL:
                                        ;               MOV     #BUFFER,R1          ;BUFFER ADDRESS
                                        ;               JSR     PC,MATCH
                                        ;               RETURN1                     ;PATTERN ADDRESS IN R4
                                        ;               RETURN2                     ;COULDN'T MATCH PATTERN

014632  010146                          MATCH:  MOV     R1,-(SP)            ;SAVE R1 ON THE STACK
014634  012704  000044                          MOV     #44,R4              ;PATTERN TABLE INDEX
014640  011601                          1$:     MOV     (SP),R1             ;RELOAD R1
014642  162704  000002                          SUB     #2,R4               ;DECREMENT INDEX
014646  001413                                  BEQ     3$                  ;BR IF  PATTERN NOT MATCH
014650  016405  002502                          MOV     STNDAT(R4),R5       ;ADDRESS OF PATTERN ADDRESS
014654  012703  000004                          MOV     #4,R3               ;NUMBER OF LOCATIONS TO CHECK
014660  022125                          2$:     CMP     (R1)+,(R5)+         ;COMPARE THE BUFFER AGAINST THE PATTERN
014662  001366                                  BNE     1$                  ;BR IF NOT EQUAL, TRY NEXT PATTERN
014664  005303                                  DEC     R3                  ;FINISHED CHECKING?
014666  001374                                  BNE     2$                  ;BR IF NOT FINISHED
014670  062704  002502                          ADD     #STNDAT,R4          ;MAKE PATTERN ADDRESS ABSOLUTE
014674  000403                                  BR      4$                  ;EXIT
014676  062766  000002  000002          3$:     ADD     #2,2(SP)            ;INCREMENT RETURN ADDRESS
014704  012601                          4$:     MOV     (SP)+,R1            ;RESTORE R1
014706  000207                                  RTS     PC                  ;RETURN

                                        ;USE ECC TO CORRECT THE DATA ERROR

014710  016037  002122  001422          ECC:    MOV     $RMBA(R0),ECSEC     ;ADDRESS OF LAST LOCN XFERED
014716  016046  002120                          MOV     $RMWC(R0),-(SP)     ;ACT WORDS XFERED (2'S COMP)
014722  066016  000020                          ADD     $WRDL(R0),(SP)      ;ADD WORDS REQUESTED
014726  005046                                  CLR     -(SP)               ;CLEAR NEXT STACK LOCN
014730  016046  000022                          MOV     $SSEC(R0),-(SP)     ;SECTOR SIZE
014734  004737  030610                          JSR     PC,LINKDV           ;DIVIDE WORDS XFERED BY SECTOR SIZE
014740  005716                                  TST     (SP)                ;PARTIAL SECTOR XFERED ?
014742  001413                                  BEQ     1$                  ;BR IF NOT
014744  006316                                  ASL     (SP)                ;CONVERT INTO NUMBER OF BYTES
014746  161637  001422                          SUB     (SP),ECSEC          ;SUBTRACT SECTOR RESIDUE
014752  126027  000024  000005                  CMPB    $CODE(R0),#5        ;WAS OP READ HEAD & DATA
014760  001007                                  BNE     2$                  ;BR IF NOT
014762  062737  000004  001422                  ADD     #4,ECSEC            ;ADD HEADER SIZE (IN BYTES) BACK IN
014770  000403                                  BR      2$                  ;GO ADJUST THE STACK POINTER
014772  162737  001000  001422          1$:     SUB     #1000,ECSEC         ;SUBTRACT SECTOR DATA FIELD SIZE (IN BYTES)
015000  062706  000004                  2$:     ADD     #4,SP               ;ADJUST THE STACK POINTER
015004  016037  002162  001420                  MOV     $RMEC1(R0),ECBIT    ;ECC POSITION COUNT
015012  005337  001420                          DEC     ECBIT               ;ADJUST THE POSITION COUNT
015016  013737  001420  001430                  MOV     ECBIT,ECWRD         ;LOAD THE WORD COUNT LOCATION
015024  042737  177760  001420                  BIC     #^C17,ECBIT         ;SAVE THE BIT OFFSET COUNT
015032  042737  000017  001430                  BIC     #17,ECWRD           ;CLEAR THE BIT OFFSET
```

```
015040  006237  001430                  ASR     ECWRD           ;CHANGE TO BYTE COUNT
015044  006237  001430                  ASR     ECWRD           ;CHANGE TO BYTE COUNT
015050  006237  001430                  ASR     ECWRD           ;CHANGE TO BYTE COUNT
015054  104414  072704                  DISPLY  ,LIN10A         ;'ERROR BURST BEGINS AT '
015060  013746  001430                  MOV     ECWRD,-(SP)     ;PUT THE WORD COUNT ON THE STACK
015064  006216                          ASR     (SP)            ;CONVERT TO WORD COUNT FOR MESSAGE
015066  004737  031524                  JSR     PC,$SB2D        ;CONVERT THE WORD COUNT
015072  004737  031124                  JSR     PC,$SUPRS       ;PRINT IT
015076  104414  072740                  DISPLY  ,LIN10B         ;' IN DATA FIELD OF ERROR SECTOR'
015102  063737  001422  001430          ADD     ECSEC,ECWRD     ;FIND THE BEGINNING OF THE ERROR BURST
015110  026037  002122  001430          CMP     $RMBA(R0),ECWRD  ;SEE IF BURST WAS IN DATA READ
015116  101002                          BHI     .+6             ;BR IF IN DATA READ
015120  000137  015436                  JMP     ECC2            ;NOT IN DATA READ - REPORT IT
015124  016037  002164  001424          MOV     $RMEC2(R0),ECMSK0  ;GET THE ERROR MASK
015132  005037  001426                  CLR     ECMSK1          ;CLEAR THE UPPER MASK WORD
015136  005737  001420          3$:     TST     ECBIT           ;BIT OFFSET EQUAL ZERO
015142  001407                          BEQ     4$              ;BR IF IT IS
015144  005337  001420                  DEC     ECBIT           ;DECREMENT THE BIT OFFSET COUNT
015150  006337  001424                  ASL     ECMSK0          ;SHIFT THE ERROR MASK
015154  006137  001426                  ROL     ECMSK1          ;SHIFT THE LOWER INTO THE UPPER
015160  000766                          BR      3$              ;CONTINUE THE SHIFT
015162  017737  164242  001434  4$:     MOV     @ECWRD,ECBAD0   ;SAVE THE INCORRECT WORD
015170  005037  001436                  CLR     ECWRD1          ;CLEAR SECOND INCORRECT WORD ADDRESS
015174  013746  001424                  MOV     ECMSK0,-(SP)    ;PUT LOWER MASK ON STACK
015200  047716  164224                  BIC     @ECWRD,(SP)     ;CLEAR ERRONEOUS ONE BITS FROM MASK
015204  043777  001424  164216          BIC     ECMSK0,@ECWRD   ;CLEAR ERRONEOUS ONE BITS FROM BAD WORD
015212  052677  164212                  BIS     (SP)+,@ECWRD    ;SET DROPPED BITS
015216  005737  001426                  TST     ECMSK1          ;DOES BURST GO INTO NEXT WORD ?
015222  001431                          BEQ     ECC1            ;BR IF BURST ONLY IN ONE WORD
015224  013737  001430  001436          MOV     ECWRD,ECWRD1    ;DUPLICATE ADDRESS
015232  062737  000002  001436          ADD     #2,ECWRD1       ;INCREMENT ERROR ADDRESS
015240  026037  002122  001436          CMP     $RMBA(R0),ECWRD1  ;IS NEXT WORD IN THE BUFFER
015246  101003                          BHI     5$              ;BR IF IT IS
015250  005037  001436                  CLR     ECWRD1          ;CLEAR 2ND WORD ADDRESS
015254  000414                          BR      ECC1            ;PRINT WORD CORRECTED
015256  017737  164154  001442  5$:     MOV     @ECWRD1,ECBAD1  ;SAVE THE SECOND BAD WORD
015264  013746  001426                  MOV     ECMSK1,-(SP)    ;PUT THE UPPER MASK ON THE STACK
015270  047716  164142                  BIC     @ECWRD1,(SP)    ;CLEAR ERRONEOUS ONE BITS FROM UPPER MASK
015274  043777  001426  164134          BIC     ECMSK1,@ECWRD1  ;CLEAR ERRONEOUS ONE BITS FROM DATA WORD
015302  052677  164130                  BIS     (SP)+,@ECWRD1   ;SET DROPPED BITS
015306  104414  073106          ECC1:   DISPLY  ,LIN10H         ;HEADER
015312  013746  001430                  MOV     ECWRD,-(SP)     ;PUT ECWRD ON THE STACK
015316  004737  023124                  JSR     PC,LINOCT       ;TYPE ECWRD
015322  104414  073270                  DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
015326  013746  001434                  MOV     ECBAD0,-(SP)    ;PUT ECBAD0 ON THE STACK
015332  004737  023124                  JSR     PC,LINOCT       ;TYPE ECBAD0
015336  104414  073270                  DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
015342  017746  164062                  MOV     @ECWRD,-(SP)    ;PUT @ECWRD ON THE STACK
015346  004737  023124                  JSR     PC,LINOCT       ;TYPE @ECWRD
015352  104414  073270                  DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
015356  005737  001436                  TST     ECWRD1          ;PRINT THE NEXT WORD ?
015362  001427                          BEQ     ECCX            ;BR IF NOT
015364  104414  001203                  DISPLY  ,$CRLF          ;CR-LF
015370  013746  001436                  MOV     ECWRD1,-(SP)    ;PUT ECWRD1 ON THE STACK
015374  004737  023124                  JSR     PC,LINOCT       ;TYPE ECWRD1
015400  104414  073270                  DISPLY  ,BLNKS2         ;TYPE 2 BLANKS
015404  013746  001442                  MOV     ECBAD1,-(SP)    ;PUT ECBAD1 ON THE STACK
```

```
015410  004737  023124                    JSR     PC,LINOCT         ;TYPE ECBAD1
015414  104414  073270                    DISPLY  ,BLNKS2           ;TYPE 2 BLANKS
015420  017746  164012                    MOV     @ECWRD1,-(SP)     ;PUT @ECWRD1 ON THE STACK
015424  004737  023124                    JSR     PC,LINOCT         ;TYPE @ECWRD1
015430  104414  073270                    DISPLY  ,BLNKS2           ;TYPE 2 BLANKS
015434  000402                            BR      ECCX              ;EXIT
015436  104414  073001            ECC2:   DISPLY  ,LIN10C           ;ERROR BURST WAS NOT TRANSFERED TO MEMORY
015442  104414  001203            ECCX:   DISPLY  ,$CRLF            ;CR-LF
015446  000207                            RTS     PC                ;RETURN

                                  ;ROUTINE TO DISPLAY THE SECTOR WHICH GAVE THE HARD ERROR

015450  032777  000010  163476    PRTBAD: BIT     #SW3,@SWR         ;PRINT THE BAD SECTOR ?
015456  001460                            BEQ     6$                ;BR IF NOT
015460  016001  002122                    MOV     $RMBA(R0),R1      ;PUT THE END ADDRESS INTO R1
015464  016046  000020                    MOV     $WRDL(R0),-(SP)   ;FIND THE BEGINNING OF THE SECTOR
015470  066016  002120                    ADD     $RMWC(R0),(SP)    ;SUBTRACT THE WORDS NOT TRANSFERED
015474  005046                            CLR     -(SP)             ;MAKE THE UPPER DIVIDEND 0
015476  016046  000022                    MOV     $SSEC(R0),-(SP)    ;DIVDE THE WORDS TRANSFERED BY THE SECTOR SIZE
015502  004737  030610                    JSR     PC,LINKDV         ;DIVIDE
015506  005716                            TST     (SP)              ;REMANDER = 0 ?
015510  001403                            BEQ     1$                ;BR IF IT IS - COMPLETE SECTOR TRANSFERED
015512  006316                            ASL     (SP)              ;CONVERT THE RESIDUAL SECTOR SIZE INTO BYTE COUNT
015514  161601                            SUB     (SP),R1           ;SUBTRACT IT FROM THE END ADDRESS
015516  000410                            BR      2$                ;FINISH THE SIZING
015520  162701  001000            1$:     SUB     #1000,R1          ;SUBTRACT FULL SECTOR SIZE FROM END ADDR
015524  126027  000024  000005            CMPB    $CODE(R0),#5      ;WAS OPERATION READ HEADER & DATA ?
015532  001002                            BNE     2$                ;BR IF NOT
015534  162701  000004                    SUB     #4,R1             ;SUBTRACK HEADER SIZE FROM ADDR
015540  062706  000004            2$:     ADD     #4,SP             ;RESTORE THE STACK POINTER
015544  104414  073173                    DISPLY  ,LIN11H           ;PRINT THE HEADER
015550  012702  000007            3$:     MOV     #7,R2             ;R2 CONTAINS THE WORDS/LINE COUNT
015554  010146                            MOV     R1,-(SP)          ;PUT THE ADDRESS ON THE STACK
015556  004737  023124                    JSR     PC,LINOCT         ;TYPE THE ADDRESS
015562  020160  002122            4$:     CMP     R1,$RMBA(R0)      ;PRINTED ALL THE SECTOR ?
015566  001412                            BEQ     5$                ;BR IF ALL PRINTED
015570  104414  073270                    DISPLY  ,BLNKS2           ;TYPE 2 BLANKS
015574  012146                            MOV     (R1)+,-(SP)       ;PUT THE DATA ON THE STACK
015576  004737  023124                    JSR     PC,LINOCT         ;TYPE THE DATA
015602  005302                            DEC     R2                ;DECREMENT THE HORIZONTAL COUNT
015604  001366                            BNE     4$                ;BR IF NOT AT THE END OF THE LINE
015606  104414  001203                    DISPLY  ,$CRLF            ;CR-LF
015612  000756                            BR      3$                ;RESTORE THE WORDS/LINE COUNT
015614  104414  001203            5$:     DISPLY  ,$CRLF            ;PRINT WHAT REMAINS IN THE BUFFER
015620  000207                    6$:     RTS     PC                ;RETURN

                                  ;ROUTINE TO DO AN RTC - DRIVE SELECTED IN R0
                                  ;CALL:
                                  ;       MOV     #DPB,R0           ;DPB ADDRESS
                                  ;       JSR     PC,RTNCTR
                                  ;       RETURN

015622  111037  066070            RTNCTR: MOVB    (R0),GENDPB       ;MOVE THE DRIVE # TO THE GENERAL DPB
015626  112737  000117  066072            MOVB    #RTC,GENDPB+$COMND  ;COMMAND CODE
015634  004037  036550            1$:     JSR     R0,RM05           ;DRIVER ENTRANCE
015640  066070                            GENDPB                    ;DPB ADDRESS FOR ORDER
015642  000774                            BR      1$                ;DRIVER DIDN'T ACCEPT ORDER
```

```
015644  000207                                   RTS     PC                      ;RETURN

                                         ;ROUTINE TO DO A RECALIBRATE - DRIVE SELECTED IN R0
                                         ;CALL:
                                         ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                         ;       JSR     PC,RECALT
                                         ;       RETURN
                                         ;OR
                                         ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                         ;       MOVB    #DRIVE,GENDPB           ;DRIVE ADDRESS
                                         ;       JSR     PC,RECALT0
                                         ;       RETURN

015646  111037  066070                   RECALT: MOVB    (R0),GENDPB             ;MOVE THE DRIVE # TO THE GENERAL DPB
015652  112737  000107  066072           RECAL0: MOVB    #RECAL,GENDPB+$COMND    ;RELCALIBRATE COMMAND
015660  005060  002124                           CLR     $RMDA(R0)               ;FAKE OUT REGISTERS FOR RECAL
015664  005060  002152                           CLR     $RMDC(R0)
015670  004037  036550                   1$:     JSR     R0,RM05                 ;DRIVER ENTRANCE
015674  066070                                   GENDPB                          ;DPB ADDRESS FOR ORDER
015676  000774                                   BR      1$                      ;DRIVER DIDN'T ACCEPT THE ORDER
015700  005737  066106                   2$:     TST     GENDPB+$STATUS          ;SEE IF FINISHED
015704  001775                                   BEQ     2$                      ;BR IF NOT FINISHED
015706  000207                                   RTS     PC                      ;RETURN

                                         ;OFFSET THE DRIVE IN R0 (OFFSET CODE PRELOADED INTO 'RMOF')
                                         ;CALL:
                                         ;       MOVB    #OFFSET,GENDPB+$FMT     ;OFFSET CODE
                                         ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                         ;       JSR     PC,OFFST
                                         ;       RETURN

015710  111037  066070                   OFFST:  MOVB    (R0),GENDPB             ;DRIVE # TO GENERAL DPB
015714  112737  000115  066072                   MOVB    #OFFSET,GENDPB+$COMND   ;COMMAND
015722  004037  036550                   1$:     JSR     R0,RM05                 ;DRIVER ENTRANCE
015726  066070                                   GENDPB                          ;DPB ADDRESS FOR ORDER
015730  000774                                   BR      1$                      ;DRIVER DIDN'T ACCEPT ORDER
015732  000207                                   RTS     PC

                                         ;UTILITY READ HEADER ROUTINE
                                         ;CALL:
                                         ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                         ;       MOV     #SECTOR,-(SP)           ;SECTOR ADDRESS
                                         ;       MOV     #TRACK,-(SP)            ;TRACK ADDRESS
                                         ;       MOV     #CYLINDER,-(SP)         ;CYLINDER ADDRESS
                                         ;       JSR     PC,READDR
                                         ;       RETURN

015734  116637  000004  066101           READHD: MOVB    4(SP),GENDPB+$TRK       ;TRACK ADDRESS
015742  116637  000006  066100                   MOVB    6(SP),GENDPB+$SEC       ;SECTOR ADDRESS
015750  016637  000002  066102                   MOV     2(SP),GENDPB+$CYL               ;CYLINDER ADDRESS
015756  111037  066070                           MOVB    (R0),GENDPB             ;DRIVE NUMBER
015762  112737  000173  066072                   MOVB    #RDHD,GENDPB+$COMND     ;COMMAND
015770  012737  177776  066074                   MOV     #-2,GENDPB+$WRDM                ;WORD CTR = 2
015776  004037  036550                   1$:     JSR     R0,RM05                 ;DRIVER ENTRANCE
016002  066070                                   GENDPB                          ;DPB ADDRESS FOR ORDER
016004  000774                                   BR      1$                      ;DRIVER DIDN'T ACCEPT COMMAND
016006  005737  066106                   2$:     TST     GENDPB+$STATUS          ;FINISHED?
```

```
016012 001775                           BEQ     2$                ;BR IF NOT
016014 011666  000006                   MOV     (SP),6(SP)        ;ADJUST STACK FOR RETURN
016020 062706  000006                   ADD     #6,SP             ;ADJUST RETRUN POINTER
016024 000207                           RTS     PC                ;RETURN

                               ;RETRY THE PRESENT OPERATION
                               ;CALL:
                               ;        MOV     #COUNT,RETRY      ;RETRY COUNT
                               ;        JSR     PC,$RETRY
                               ;        RETURN1                   ;RETRY UNSUCESSFUL
                               ;        RETURN2                   ;SUCESSFUL RETRY
                               ;                                  ;NOTE: IF A DIFFERENT ERROR OCCURS DURING
                               ;                                  ;RETRY, THE ROUTINE EXITS TO 'ERPRC1'

016026 004737  017024          $RETRY:  JSR     PC,GODRIV         ;RE-START ORDER
016032 005760  000016          1$:      TST     $STATUS(R0)       ;ORDER FINISHED?
016036 001775                           BEQ     1$                ;BR IF NOT
016040 100405                           BMI     2$                ;BR IF ERROR
016042 105237  001353                   INCB    RETRY+1           ;INCREMENT RETRY COUNT
016046 062716  000002                   ADD     #2,(SP)           ;INCREMENT RETURN
016052 000425                           BR      5$                ;GO TO EXIT
016054 032760  000200  000016  2$:      BIT     #BIT7,$STATUS(R0)   ;DID ORDER TERMINATE NORMALLY ?
016062 001430                           BEQ     7$                ;BR IF NOT
016064 005737  001350                   TST     MASK              ;IS ERROR MASK 0 ?
016070 001004                           BNE     3$                ;BR IF NOT
016072 005760  002132                   TST     $RMER1(R0)        ;MAKE SURE THAT THE DRIVE ERROR REG IS CLEAR
016076 001014                           BNE     6$                ;BR IF NOT
016100 000404                           BR      4$                ;CONTINUE RETRY
016102 033760  001350  002132  3$:      BIT     MASK,$RMER1(R0)   ;SAME ERROR?
016110 001407                           BEQ     6$                ;BR IF NOT
016112 105237  001353          4$:      INCB    RETRY+1           ;INCREMENT RETRY COUNT
016116 123737  001352  001353           CMPB    RETRY,RETRY+1     ;DONE ?
016124 001340                           BNE     $RETRY            ;BR IF NOT DONE
016126 000207                  5$:      RTS     PC                ;RETURN
016130 004737  023112          6$:      JSR     PC,LINE8          ;REPORT DIFFERENT ERROR
016134 004737  022654                   JSR     PC,LINE7          ;PRINT LINE 7
016140 005726                           TST     (SP)+             ;ADJUST STACK POINTER FOR DIRECT RETURN
016142 000207                           RTS     PC                ;RETURN
016144 104414  072455          7$:      DISPLY  ,LIN8M            ;'DIFFERENT ERROR DURING RETRY'
016150 000137  007266                   JMP     ERPRC1            ;REPORT THE ERROR

                               ;ROUTINE TO UPDATE THE PERFORMANCE SUMMARY STATISTICS
                               ;CALL:
                               ;        MOV     #DPB,R0           ;DPB ADDRESS
                               ;        JSR     PC,STATIS
                               ;        RETURN

016154 032760  000300  000016  STATIS:  BIT     #BIT07!BIT06,$STATUS(R0) ;CHECK FOR DATA TERMINATION
016162 001451                           BEQ     3$                ;BR IF NOT DATA TERMINATION
016164 016037  002122  016310           MOV     $RMBA(R0),FACTOR  ;STORE THE FINAL BUFFER ADDRESS
016172 166037  000006  016310           SUB     $BUF(R0),FACTOR   ;SUBTRACT THE INITIAL ADDRESS
016200 001431                           BEQ     2$                ;BR IF NO DATA TRANSFER
016202 006237  016310                   ASR     FACTOR            ;CONVERT TO A WORD COUNT
016206 063760  016310  000046           ADD     FACTOR,$TRANS(R0)   ;UPDATE WORD COUNT
016214 005560  000050                   ADC     $TRANS+2(R0)      ;ADD ANY CARRY
016220 132760  000002  000024           BITB    #BIT01,$CODE(R0)   ;SEE IF ORDER READ OR WRITE
016226 001016                           BNE     2$                ;BRANCH IF ORDER WRITE
```

```
016230  126027  000024  000001          CMPB    $CODE(R0),#1        ;PRESENT OPERATION AN AUTOMATIC WRITE CHECK ?
016236  101005                          BHI     1$                  ;BR IF NOT
016240  066060  000020  000046          ADD     $WRDL(R0),$TRANS(R0)   ;ADD WORDS WRITTEN
016246  005560  000050                  ADC     $TRANS+2(R0)        ;ADD A CARRY
016252  063760  016310  000052  1$:     ADD     FACTOR,$READ(R0)    ;UPDATE THE READ WORD COUNT
016260  005560  000054                  ADC     $READ+2(R0)         ;ADD ANY CARRY
016264  026060  000012  002152  2$:     CMP     $CYL(R0),$RMDC(R0)   ;DID MID-TRANSFER SEEK OCCUR
016272  001405                          BEQ     3$                  ;BR IF NOT
016274  062760  000001  000042          ADD     #1,$POSIT(R0)       ;INCREMENT SEEK COUNT
016302  005560  000044                  ADC     $POSIT+2(R0)        ;ADD CARRY TO UPPER WORD
016306  000207                  3$:     RTS     PC

016310  000000                  FACTOR: .WORD   0                   ;USED FOR WORDS TRANSFERED

                                ;ROUTINE TO GET A BUFFER
                                ;CALL:
                                ;       MOV     #DPB,R0             ;DPB ADDRESS
                                ;       CLR     -(SP)               ;CLEAR THE STACK
                                ;       JSR     PC,GETBUF
                                ;       RETURN                      ;BUFFER ADDRESS WILL BE ON THE STACK
                                ;                                   ;STACK WILL BE ZERO IF NO BUFFER AVAILABLE

016312  010146                  GETBUF: MOV     R1,-(SP)            ;SAVE R1
016314  010246                          MOV     R2,-(SP)            ;SAVE R2
016316  010346                          MOV     R3,-(SP)            ;SAVE R3
016320  013702  001704                  MOV     BUFTBL,R2           ;NUMBER OF SEPARATE BUFFERS
016324  001444                          BEQ     6$                  ;BR IF NONE AVAILABLE
016326  012701  001706                  MOV     #BUFTBL+2,R1        ;FIRST ADDRESS OF ALLOCATION TABLE
016332  026061  000020  000002  1$:     CMP     $WRDL(R0),2(R1)     ;SEE IF THERE IS A BLOCK LARGE ENOUGH
016340  101405                          BLOS    3$                  ;BRANCH IF IT IS
016342  005302                          DEC     R2                  ;DECREMENT TABLE COUNT
016344  001434                          BEQ     6$                  ;BR IF THROUGH TABLE
016346  062701  000004                  ADD     #4,R1               ;INCREMENT TABLE POINTER
016352  000767                          BR      1$                  ;CONTINUE LOOKING
016354  011166  000010          3$:     MOV     (R1),10(SP)         ;BUFFER ADDRESS TO STACK
016360  166061  000020  000002          SUB     $WRDL(R0),2(R1)     ;ADJUST BUFFER SIZE
016366  001407                          BEQ     4$                  ;L  IF DIFFERENCE IS ZERO
016370  006360  000020                  ASL     $WRDL(R0)           ;CONVERT # WORDS TO BYTES
016374  066011  000020                  ADD     $WRDL(R0),(R1)      ;MAKE NEW STARTING ADDRESS
016400  006260  000020                  ASR     $WRDL(R0)           ;RETURN # BYTES TO WORDS
016404  000414                          BR      6$                  ;RETURN
016406  005337  001704          4$:     DEC     BUFTBL              ;DECREMENT ENTRIES COUNT
016412  001411                          BEQ     6$                  ;BR IF ALLOCATION TABLE EMPTY
016414  005302                          DEC     R2                  ;DECREMENT TABLE COUNT
016416  001407                          BEQ     6$                  ;BR IF ITEM WERE LAST ENTRY
016420  010103                          MOV     R1,R3               ;MOVE TABLE POINTER
016422  062703  000004                  ADD     #4,R3               ;POINT TO NEXT ENTRY
016426  012321                  5$:     MOV     (R3)+,(R1)+         ;MOVE ITEMS
016430  012321                          MOV     (R3)+,(R1)+
016432  005302                          DEC     R2                  ;DECREMENT TABLE COUNT
016434  001374                          BNE     5$                  ;CONTINUE IF NOT AT END OF TABLE
016436  012603                  6$:     MOV     (SP)+,R3            ;RESTORE R3
016440  012602                          MOV     (SP)+,R2            ;RESTORE R2
016442  012601                          MOV     (SP)+,R1            ;RESTORE R1
016444  000207                          RTS     PC                  ;RETURN
```

```
                                                ;ROUTINE TO PUT BUFFER BACK IN TABLE
                                                ;CALL:
                                                ;        MOV     #DPB,R0           ;DPB ADDRESS
                                                ;        JSR     PC,RELBUF
                                                ;        RETURN

016446  010146                                  RELBUF: MOV     R1,-(SP)          ;SAVE R1
016450  010246                                          MOV     R2,-(SP)          ;SAVE R2
016452  010446                                          MOV     R4,-(SP)          ;SAVE R4
016454  010546                                          MOV     R5,-(SP)          ;SAVE R5
016456  012701  001706                                  MOV     #BUFTBL+2,R1      ;BEGINNING OF TABLE
016462  013702  001704                                  MOV     BUFTBL,R2         ;ENTRY COUNT
016466  001424                                          BEQ     2$                ;BR IF EMPTY TABLE
016470  016003  000020                                  MOV     $WRDL(R0),R3      ;TRIAL ADDRESS
016474  006303                                          ASL     R3                ;CHANGE TO BYTE COUNT
016476  066003  000006                                  ADD     $BUF(R0),R3       ;ADDRESS OF HIGHER ADJACENT BLOCK
016502  021103                                  1$:     CMP     (R1),R3           ;UPPER ADJACENT BLOCK
016504  001424                                          BEQ     4$                ;BR IF YES
016506  062701  000004                                  ADD     #4,R1             ;INCREMENT POINTER
016512  005302                                          DEC     R2                ;DECREMENT ENTRY COUNT
016514  001372                                          BNE     1$                ;CONTINUE SEARCHING
016516  016011  000006                                  MOV     $BUF(R0),(R1)     ;PUT THE BUFFER BLOCK INTO THE TABLE
016522  016061  000020  000002                          MOV     $WRDL(R0),2(R1)    ;BLOCK SIZE
016530  005237  001704                                  INC     BUFTBL            ;INCREMENT ENTRY COUNT
016534  005202                                          INC     R2                ;INCREMENT R2 FOR USE LATER
016536  000414                                          BR      5$                ;SEE IF A LOWER ADJACENT BLOCK IS IN THE TABLE
016540  016021  000006                          2$:     MOV     $BUF(R0),(R1)+    ;BLOCK ADDRESS TO TABLE
016544  016021  000020                                  MOV     $WRDL(R0),(R1)+   ;SIZE TO TABLE
016550  005237  001704                                  INC     BUFTBL            ;INCREMENT ENTRY COUNT
016554  000443                                          BR      10$               ;EXIT
016556  016011  000006                          4$:     MOV     $BUF(R0),(R1)     ;RELEASED BUFFER IS LOWER ADJACENT
016562  066061  000020  000002                          ADD     $WRDL(R0),2(R1)   ;INCREMENTED SIZE
016570  010246                                  5$:     MOV     R2,-(SP)          ;SAVE R2
016572  013702  001704                                  MOV     BUFTBL,R2         ;ENTRY COUNT
016576  012705  001706                                  MOV     #BUFTBL+2,R5      ;BEGINNING OF TABLE
016602  016504  000002                          6$:     MOV     2(R5),R4          ;BLOCK SIZE (IN WORDS)
016606  006304                                          ASL     R4                ;CHANGE TO BYTE COUNT
016610  061504                                          ADD     (R5),R4           ;ADD BLOCK BEGINNING ADDRESS
016612  020411                                          CMP     R4,(R1)           ;R1 STILL POINTS TO INSERTED ENTRY
016614  001406                                          BEQ     8$                ;LOWER ADJACENT IN TABLE
016616  062705  000004                                  ADD     #4,R5             ;INCREMENT POINTER
016622  005302                                          DEC     R2                ;DECREMENT ENTRY COUNT
016624  001366                                          BNE     6$                ;CONTINUE LOOKING
016626  005726                                          TST     (SP)+             ;RESTORE STACK POINTER
016630  000415                                          BR      10$               ;END
016632  012602                                  8$:     MOV     (SP)+,R2          ;RESTORE R2
016634  066165  000002  000002                          ADD     2(R1),2(R5)       ;INCREMENT LOWER BLOCK LENGTH
016642  005337  001704                                  DEC     BUFTBL            ;DECREMENT ENTRY COUNT
016646  010105                                          MOV     R1,R5             ;GET READY TO COMPRESS
016650  062705  000004                                  ADD     #4,R5             ;INCREMENT TO NEXT ENTRY
016654  012521                                  9$:     MOV     (R5)+,(R1)+       ;COMPRESS TABLE
016656  012521                                          MOV     (R5)+,(R1)+       ;MOVE SIZE FIELD DOWN
016660  005302                                          DEC     R2                ;DECREMENT ENTRY COUNT
016662  001374                                          BNE     9$                ;BR IF NOT FINISHED
016664  012605                                  10$:    MOV     (SP)+,R5          ;RESTORE R5
016666  012604                                          MOV     (SP)+,R4          ;RESTORE R4
016670  012602                                          MOV     (SP)+,R2          ;RESTORE R2
```

```
016672  012601                          MOV     (SP)+,R1        ;RESTORE R1
016674  000207                          RTS     PC              ;RETURN


                                ;FILL THE ASSIGNED BUFFER (IF WRITE OR WRITE CHECK ORDER)
                                ;CALL:
                                ;       MOV     #DPB,R0         ;DPB ADDRESS
                                ;       MOV     #BUFADR,$BUF(R0)   ;LOAD BUFFER ADDRESS INTO THE DPB
                                ;       MOVB    #PATTERN,$PATTC(R0)   ;PATTERN CODE
                                ;       JSR     PC,FILBUF
                                ;       RETURN

016676  104412                  FILBUF: SAVREG                  ;SAVE THE REGISTERS
016700  132760  000004  000024          BITB    #BIT02,$CODE(R0)   ;SEE IF READ ORDER
016706  001044                          BNE     4$              ;BR IF READ
016710  016001  000006          1$:     MOV     $BUF(R0),R1     ;BUFFER ADDRESS
016714  016002  000020                  MOV     $WRDL(R0),R2    ;POSITIVE WORD COUNT
016720  132760  000001  000024          BITB    #BIT00,$CODE(R0)   ;SEE IF WRITE HEADER TYPE ORDER
016726  001413                          BEQ     2$              ;BR IF NOT
016730  016011  000012                  MOV     $CYL(R0),(R1)   ;CYLINDER ADDRESS
016734  052711  010000                  BIS     #BIT12,(R1)     ;SET FMT22 BIT
016740  052721  140000                  BIS     #140000,(R1)+   ;SET MFG AND USER BITS
016744  016021  000010                  MOV     $SEC(R0),(R1)+  ;MOVE SECTOR & TRACK
016750  162702  000002                  SUB     #2,R2           ;ADJUST THE WORD COUNT
016754  003421                          BLE     4$              ;BR IF END OF PATTERN
016756  005004                  2$:     CLR     R4              ;CLEAR R4
016760  116004  000030                  MOVB    $PATTC(R0),R4   ;RELATIVE PATTERN ADDRESS
016764  016405  002502                  MOV     STNDAT(R4),R5   ;PATTERN ADDRESS
016770  012703  000020                  MOV     #20,R3          ;PATTERN COUNT
016774  012521                  3$:     MOV     (R5)+,(R1)+     ;MOVE THE PATTERN INTO THE BUFFER
016776  005302                          DEC     R2              ;DECREMENT THE WORD COUNT
017000  003407                          BLE     4$              ;BR IF DONE (WORD COUNT = 0)
017002  005303                          DEC     R3              ;DECREMENT THE PATTERN COUNT
017004  001373                          BNE     3$              ;BR IF MORE PATTERN
017006  012703  000020                  MOV     #20,R3          ;RESTORE PATTERN COUNT
017012  016405  002502                  MOV     STNDAT(R4),R5   ;RESTORE THE ADDRESS
017016  000766                          BR      3$              ;CONTINUE DISTRIBUTING THE PATTERN
017020  104413                  4$:     RESREG                  ;RESTORE THE REGISTERS
017022  000207                          RTS     PC              ;RETURN

                                ;START THE ORDER FOR THE DPB IN R0
                                ;CALL:
                                ;       MOV     #DPB,R0         ;DPB ADDRESS
                                ;       JSR     PC,GODRIV
                                ;       RETURN

017024  010046                  GODRIV: MOV     R0,-(SP)        ;SAVE R0
017026  010037  017036                  MOV     R0,2$           ;CURRENT DPB ADDRESS
017032  004037  036550          1$:     JSR     R0,RM05         ;CALL THE DRIVE HANDLER
017036  000000                  2$:     .WORD   0               ;DRIVE BLOCK ADDRESS GOES HERE
017040  000000                          HALT                    ;DRIVER REJECTED REQUEST
017042  012600                          MOV     (SP)+,R0        ;RESTORE R0
017044  062760  000001  000036          ADD     #1,$OPERC(R0)   ;INCREMENT THE OPERATION COUNT
017052  005560  000040                  ADC     $OPERC+2(R0)
017056  026060  000034  000012          CMP     $PREVA+2(R0),$CYL(R0)   ;DID ORDER REQUIRE A CYLINDER CHANGE
017064  001405                          BEQ     3$              ;BR IF NOT
017066  062760  000001  000042          ADD     #1,$POSIT(R0)   ;INCREMENT SEEK COUNT
```

```
017074  005560  000044                  ADC     $POSIT+2(R0)    ;ADD ANY CARRY
017100  000207                  3$:     RTS     PC


                                ;GENERATE PARAMETERS FOR THE OPERATION
                                ;CALL:
                                ;       MOV     #DPB,R0         ;DPB ADDRESS
                                ;       JSR     PC,SELPAR
                                ;       RETURN

017102  004737  035014          SELPAR: JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
017106  032777  000001  162040          BIT     #SW0,@SWR       ;SEE IF SW0 SET
017114  001012                          BNE     2$              ;BR IF SET - READ ONLY
017116  012705  000010          1$:     MOV     #10,R5          ;READ/WRITE SELECTION DIVISOR
017122  004737  030562                  JSR     PC,GETREM       ;GET SELECTION VALUE
017126  020537  001504                  CMP     R5,RATIO        ;DETERMINE IF READ OR WRITE
017132  103003                          BHIS    2$              ;BR IF READ
017134  004737  017766                  JSR     PC,RANWRT       ;SELECT A WRITE ORDER
017140  000410                          BR      THEAD           ;SELECT ADDRESS
017142  013705  035114          2$:     MOV     $LONUM,R5       ;SELECT READ OPERATION CODE
017146  042705  177776                  BIC     #^C1,R5         ;MASK OUT ALL BUT BIT 0
017152  062705  000004                  ADD     #4,R5           ;TABLE OFFSET FOR READ CODE
017156  110560  000074                  MOVB    R5,$NCODE(R0)   ;ORDER SELECTION CODE TO CONTROL BLOCK
017162  005737  001516          THEAD:  TST     HEADER          ;ENABLE RANDOM ADDRESS SELECT ?
017166  001425                          BEQ     RANSEC          ;YES
017170  016060  002124  000076          MOV     $RMDA(R0),$NSEC(R0)     ;SECTOR AND TRACK
017176  016060  002152  000100          MOV     $RMDC(R0),$NCYL(R0) ;CYLINDER NUMBER
017204  026060  000100  000106          CMP     $NCYL(R0),MAXCYL(R0) ;OVER MAX CYLINDER #  ?
017212  103521                          BLO     XWCNT           ;NO
017214  016060  000110  000100          MOV     MINCYL(R0),$NCYL(R0) ;RESET CYLINDER NUMBER
017222  116060  000120  000076          MOVB    MINSEC(R0),$NSEC(R0) ;RESET SECTOR NUMBER
017230  116060  000114  000077          MOVB    MINTRK(R0),$NTRK(R0) ;RESET TRACK NUMBER
017236  000137  017456                  JMP     XWCNT           ;SELECT BUFFER SIZE

                                ;GENERATE A RANDOM SECTOR ADDRESS BETWEEN VALUES 'MINSEC' & 'MAXSEC'

017242  016005  000116          RANSEC: MOV     MAXSEC(R0),R5   ;GET MAXIMUM SECTOR ADDRESS
017246  026005  000120                  CMP     MINSEC(R0),R5   ;'MINSEC' AND 'MAXSEC' THE SAME ?
017252  001417                          BEQ     2$              ;BR IF THEY ARE
017254  166005  000120                  SUB     MINSEC(R0),R5   ;SUBTRACT MINIMUM SECTOR ADDRESS
017260  100002                          BPL     1$              ;BR IF MAX LARGER THAN MIN
017262  062705  000040                  ADD     #32.,R5         ;CORRECT THE NUMBER
017266  005205                  1$:     INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
017270  004737  030562                  JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
017274  066005  000120                  ADD     MINSEC(R0),R5   ;NEW ADDRESS
017300  020527  000037                  CMP     R5,#31.         ;IS VALUE TOO LARGE ?
017304  101402                          BLOS    2$              ;BR IF NOT
017306  162705  000040                  SUB     #32.,R5         ;CORRECT VALUE
017312  110560  000076          2$:     MOVB    R5,$NSEC(R0)    ;STORE SECTOR ADDRESS IN DPB

                                ;GENERATE A RANDOM TRACK ADDRESS BETWEEN VALUES 'MINTRK' & 'MAXTRK'

017316  004737  026546          RANTRK: JSR     PC,GETLMT       ;GET ADDRESS LIMITS
017322  016005  000112                  MOV     MAXTRK(R0),R5   ;GET MAXIMUM TRACK ADDRESS
017326  026005  000114                  CMP     MINTRK(R0),R5   ;'MINTRK' AND 'MAXTRK' THE SAME ?
017332  001421                          BEQ     3$              ;BR IF THEY ARE
017334  166005  000114                  SUB     MINTRK(R0),R5   ;SUBTRACT MINIMUM TRACK ADDRESS
017340  100003                          BPL     2$              ;BR IF MAX LARGER THAN MIN
```

```
 1560 017342  063705  001450                    ADD     TRKLMT,R5           ;CORRECT THE NUMBER
 1561 017346  005205                            INC     R5
 1562 017350  005205                    2$:     INC     R5                  ;INCREMENT DIFFERENCE TO USE AS DIVISOR
 1563 017352  004737  030562                    JSR     PC,GETREM           ;GET THE RANDOM AUGMENT
 1564 017356  066005  000114                    ADD     MINTRK(R0),R5       ;NEW TRACK ADDRESS
 1565 017362  020537  001450                    CMP     R5,TRKLMT           ;IS VALUE TOO LARGE ?
 1566 017366  003403                            BLE     3$                  ;BR IF NO
 1567 017370  163705  001450                    SUB     TRKLMT,R5           ;CORRECT VALUE
 1568 017374  005305                            DEC     R5
 1569 017376  110560  000077            3$:     MOVB    R5,$NTRK(R0)        ;STORE TRACK ADDRESS IN DPB
 1570
 1571                                   ;GENERATE A RANDOM CYLINDER ADDRESS BETWEEN VALUES 'MINCYL' & 'MAXCYL'
 1572
 1573 017402  016005  000106            RANCYL: MOV     MAXCYL(R0),R5       ;GET MAXIMUM CYLINDER ADDRESS
 1574 017406  026005  000110                    CMP     MINCYL(R0),R5       ;'MINCYL' AND 'MAXCYL' THE SAME ?
 1575 017412  001417                            BEQ     2$                  ;BR IF THEY ARE
 1576 017414  166005  000110                    SUB     MINCYL(R0),R5       ;SUBTRACT MINIMUM CYLINDER ADDRESS
 1577 017420  100002                            BPL     1$                  ;BR IF MAX LARGER THAN MIN
 1578 017422  062705  001466                    ADD     #822.,R5            ;CORRECT THE NUMBER
 1579 017426  005205                    1$:     INC     R5                  ;INCREMENT DIFFERENCE TO USE AS DIVISOR
 1580 017430  004737  030562                    JSR     PC,GETREM           ;GET THE RANDOM AUGMENT
 1581 017434  066005  000110                    ADD     MINCYL(R0),R5       ;NEW CYLINDER ADDRESS
 1582 017440  020527  001465                    CMP     R5,#821.            ;IS VALUE TOO LARGE ?
 1583 017444  003402                            BLE     2$                  ;BR IF NO
 1584 017446  162705  001466                    SUB     #822.,R5            ;CORRECT VALUE
 1585 017452  010560  000100            2$:     MOV     R5,$NCYL(R0)        ;STORE CYLINDER ADDRESS IN DPB
 1586
 1587 017456  132760  000001  000074    XWCNT:  BITB    #BIT00,$NCODE(R0) ;HEADER OPERATION INVOKED ?
 1588 017464  001414                            BEQ     RANSIZ              ;NO
 1589 017466  012760  000402  000102            MOV     #258.,$NWRDL(R0)   ;CHANGE WORD LENGTH TO 258 FOR WRTHD ORDER
 1590 017474  122760  000005  000074    3$:     CMPB    #5,$NCODE(R0)       ;READ HEADER AND DATA ?
 1591 017502  001461                            BEQ     RANXIT              ;YES
 1592 017504  004537  017656                    JSR     R5,CHKADR           ;IF WRITE HEAD AND DATA COMMAND
 1593                                                                       ;AVOID WRITING BAD SPOT HEADER
 1594 017510  000452                            BR      RANPAT              ;BRANCH IF NOT ON BAD SPOT
 1595 017512  000137  017102                    JMP     SELPAR              ;SELECT THE PARAMETERS AGAIN
 1596
 1597                                   ;GENERATE A RANDOM BUFFER LENGTH BETWEEN 4 & THE VALUE IN 'MAXDL'
 1598
 1599 017516  013705  001466            RANSIZ: MOV     MAXDL,R5            ;GET BUFFER SIZE
 1600 017522  005737  001502                    TST     WCSEL               ;SELECT A RANDOM WORD COUNT ?
 1601 017526  001010                            BNE     1$                  ;BR IF NOT
 1602 017530  005205                            INC     R5                  ;INCREMENT THE MAXIMUM SIZE
 1603 017532  004737  030562                    JSR     PC,GETREM           ;DIVIDE BY MAX VALUE
 1604 017536  005705                            TST     R5                  ;IS THE REMAINDER 0 ?
 1605 017540  001003                            BNE     1$                  ;NOT 0, CONTINUE
 1606 017542  004737  035014                    JSR     PC,$RAND            ;CYCLE THE RANDOM NUMBER GENERATOR
 1607 017546  000763                            BR      RANSIZ              ;TRY AGAIN
 1608 017550  022705  000004            1$:     CMP     #4,R5               ;LESS THAN 4 ?
 1609 017554  003403                            BLE     2$                  ;NO
 1610 017556  012705  000004                    MOV     #4,R5               ;SET SIZE TO 4
 1611 017562  000405                            BR      3$
 1612 017564  023705  001466            2$:     CMP     MAXDL,R5            ;LARGE THAN MAX ALLOWED ?
 1613 017570  101002                            BHI     3$                  ;NO
 1614 017572  013705  001466                    MOV     MAXDL,R5            ;RESET COUNTER
 1615 017576  132760  000004  000074    3$:     BITB    #BIT02,$NCODE(R0) ;READ OPERATION ?
 1616 017604  001006                            BNE     4$                  ;YES
```

```
017606  042705  000377                   BIC     #377,R5             ;SECTOR BOUNDARY FOR WRITE OP
017612  005705                           TST     R5                  ;NONE
017614  003002                           BGT     4$                  ;NO
017616  012705  000400                   MOV     #256.,R5            ;AT LEAST ONE SECTOR
017622  010560  000102          4$:      MOV     R5,$NWRDL(R0)       ;WORD COUNT
017626  132760  000004  000074           BITB    #BIT02,$NCODE(R0)  ;READ OP ?
017634  001004                           BNE     RANXIT              ;YES

                                ;GET A RANDOM PATTERN NUMBER
017636  004737  020022          RANPAT:  JSR     PC,GETPAT           ;GET PATTERN CODE
017642  110560  000075                   MOVB    R5,$NPATC(R0)       ;MOVE PATTERN CODE TO CONTROL BLOCK
017646  012760  177777  000104  RANXIT:  MOV     #-1,$NEXT(R0)      ;SET PARAMETERS SELECTED INDICATOR
017654  000207                           RTS     PC                  ;RETURN

                                ;ROUTINE TO CHECK THE SELECTED ADDRESS IS NOT ON THE BAD SPOT
                                ;CALLING SEQ
                                ;        MOV     #DPB,R0             ;DPB ADDRESS
                                ;        JSR     R5,CHKADR
                                ;        RET1                        NORMAL RETURN
                                ;        RET2                        ERROR RET
                                ;
                                ;R0 = DPB ADDRESS BEFORE CALLING THE ROUTINE

017656                          CHKADR:
017656  010246                           MOV     R2,-(SP)            ;;PUSH R2 ON STACK
017660  010346                           MOV     R3,-(SP)            ;;PUSH R3 ON STACK
017662  010002                           MOV     R0,R2               ;TABLE ADDRESS
017664  062702  000124                   ADD     #$BDSEC,R2          ;DBP ADDRESS + TABLE ADDRES OFFSET
017670  022712  177777          1$:      CMP     #-1,(R2)            ;EMPTY ENTRY OR TERMINATOR ?
017674  001431                           BEQ     5$                  ;BRANCH IF SO
017676  026012  000100                   CMP     $NCYL(R0),(R2)      ;ON THE SAME CYLINDER ?
017702  001023                           BNE     4$                  ;BRANCH IF NOT
017704  122762  177777  000003           CMPB    #-1,3(R2)           ;ON ALL TRACKS ?
017712  001404                           BEQ     2$                  ;BR IF YES
017714  126062  000011  000003           CMPB    $TRK(R0),3(R2)      ;ON THE SAME TRACK ?
017722  001013                           BNE     4$                  ;BRANCH IF NOT
017724  122762  177777  000002  2$:      CMPB    #-1,2(R2)           ;ON ALL SECTORS ?
017732  001404                           BEQ     3$                  ;BR IF YES
017734  126062  000010  000002           CMPB    $SEC(R0),2(R2)      ;ON THE SAME SECTOR ?
017742  001003                           BNE     4$                  ;BRANCH IF NOT
017744  062705  000002          3$:      ADD     #2,R5               ;FOUND BAD ADDRESS(ES) IN TABLE
017750  000403                           BR      5$                  ;EXIT
017752  062702  000004          4$:      ADD     #4,R2               ;ADJUST TO NEXT TABLE ENTRY
017756  000744                           BR      1$                  ;LOOP BACK
017760                          5$:
017760  012603                           MOV     (SP)+,R3            ;;POP STACK INTO R3
017762  012602                           MOV     (SP)+,R2            ;;POP STACK INTO R2
017764  000205                           RTS     R5                  ;RETURN

                                ;ROUTINE TO SELECT A WRITE (OR WRITE HEAD AND DATA) OPERATION

017766  012705  000002          RANWRT:  MOV     #2,R5               ;SET WRITE DATA COMMAND
017772  005737  001500                   TST     FORMAT              ;ALLOW FORMAT OPERATION ?
017776  001406                           BEQ     1$                  ;NO
020000  122760  000004  000024           CMPB    #4,$CODE(R0)        ;PREVIOUS A READ DATA COMMAND ?
020006  001002                           BNE     1$                  ;NO
020010  012705  000003                   MOV     #3,R5               ;SET A WRITE HEAD AND DATA COMMAND
```

```
020014  110560  000074          1$:     MOVB    R5,$NCODE(R0)           ;SELECT THE WRITE COMMAND
020020  000207                          RTS     PC                      ;EXIT

                                ;ROUTINE TO SELECT A PATTERN

020022  012705  000020          GETPAT: MOV     #20,R5                  ;SELECT PATTERN
020026  005737  001514                  TST     PATTEN                  ;ENABLE RANDOM PATTERN SELECTION ?
020032  001403                          BEQ     2$                      ;YES
020034  013705  001514                  MOV     PATTEN,R5               ;USE INDEXED PATTERN
020040  000407                          BR      1$                      ;NO
020042  004737  030562          2$:     JSR     PC,GETREM               ;GET CODE
020046  005705                          TST     R5                      ;WAS PATTERN ZERO SELECTED ?
020050  001003                          BNE     1$                      ;BR IF NOT ZERO
020052  004737  035014                  JSR     PC,$RAND                ;CYCLE THE RANDOM NUMBER GENERATOR
020056  000761                          BR      GETPAT                  ;TRY AGAIN
020060  006305                  1$:     ASL     R5                      ;MAKE CODE INTO TABLE INDEX
020062  000207                          RTS     PC

                                ;ROUTINE TO GET THE PREVIOUSLY SELECTED PARAMETER VALUES
                                ;CALL:
                                ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                ;       JSR     PC,SELPAR               ;SELECT THE PARAMETERS
                                ;       JSR     PC,GETPAR
                                ;       RETURN

020064  010546                  GETPAR: MOV     R5,-(SP)                ;SAVE R5
020066  116060  002116  000027          MOVB    $RMCS1(R0),$PREVO(R0)   ;SAVE CURRENT PARAMETERS
020074  142760  177701  000027          BICB    #^C76,$PREVO(R0)                ;STRIP GO,AND IE BITS
020102  032760  000006  000074          BIT     #6,$NCODE(R0)           ;SEE IF NEXT OPERATION IS READ OR WRITE
020110  001007                          BNE     1$                      ;BR IF EITHER
020112  016060  000012  000034          MOV     $CYL(R0),$PREVA+2(R0)   ;SAVE STARTING CYLINDER
020120  016060  000010  000032          MOV     $SEC(R0),$PREVA(R0)     ;SAVE STARTING SECTOR AND TRACK
020126  000410                          BR      22$
020130  004737  023176          1$:     JSR     PC,READDR               ;GET THE DECREMENTED SECTOR AND TRACK ADDRESSES
020134  012660  000034                  MOV     (SP)+,$PREVA+2(R0)              ;CYLINDER ADDRESS
020140  112660  000033                  MOVB    (SP)+,$PREVA+1(R0)              ;TRACK ADDRESS
020144  112660  000032                  MOVB    (SP)+,$PREVA(R0)        ;SECTOR ADDRESS
020150  032777  000100  160776  22$:    BIT     #SW06,@SWR              ;SWITCH 6 SET ?
020156  001073                          BNE     5$                      ;BR IF SET
020160  116060  000074  000024          MOVB    $NCODE(R0),$CODE(R0)    ;LOGICAL CODE FOR OPERATION
020166  116005  000074                  MOVB    $NCODE(R0),R5           ;LOAD R5 FOR USE AS TABLE INDEX
020172  116560  002126  000002          MOVB    COMTBL(R5),$COMND(R0)   ;COMMAND CODE
020200  122760  000151  000002          CMPB    #151,$COMND(R0)         ;WRITE CHECK DATA COMMAND ?
020206  001013                          BNE     3$                      ;NO,DO NOT CARE
020210  122760  000060  000027          CMPB    #60,$PREVO(R0)          ;PREVIOUS A WRITE DATA COMMAND ?
020216  001420                          BEQ     4$                      ;YES,O K
020220  112760  000171  000002  2$:     MOVB    #171,$COMND(R0)         ;CHANG TO READ DATA COMMAND
020226  112760  000004  000024          MOVB    #4,$CODE(R0)            ;CODE NUMBER CHANGED TO READ DATA
020234  000411                          BR      4$                      ;EXIT
020236  122760  000153  000002  3$:     CMPB    #153,$COMND(R0)         ;WRITE CHECK HEAD AND DATA COMMAND ?
020244  001005                          BNE     4$                      ;NO,THEN EXIT
020246  122760  000062  000027          CMPB    #62,$PREVO(R0)          ;PREVIOUS A WRITE HEAD AND DATA COMMAND?
020254  001401                          BEQ     4$                      ;YES,EXIT
020256  000760                          BR      2$                      ;SET TO READ DATA COMMAND
020260                          4$:
020260  116060  000075  000030          MOVB    $NPATC(R0),$PATTC(R0)   ;PATTERN CODE
020266  016060  000076  000010          MOV     $NSEC(R0),$SEC(R0)      ;TRACK AND SECTOR ADDRESSES
```

```
020274  016060  000100  000012           MOV     $NCYL(R0),$CYL(R0)   ;CYLINDER ADDRESS
020302  016060  000102  000020           MOV     $NWRDL(R0),$WRDL(R0)   ;BUFFER SIZE
020310  016060  000102  000004           MOV     $NWRDL(R0),$WRDM(R0)   ;WORD COUNT
020316  005460  000004                   NEG     $WRDM(R0)            ;COMPLEMENT IT
020322  012760  000400  000022           MOV     #256.,$SSEC(R0)  ;INITIAL VALUE OF SECTOR SIZE
020330  032760  000001  000024           BIT     #1,$CODE(R0)     ;HEADER OPERATION ?
020336  001403                           BEQ     5$               ;BR IF NOT
020340  062760  000002  000022           ADD     #2,$SSEC(R0)     ;ADD HEADER SIZE
020346  005060  000104          5$:      CLR     $NEXT(R0)        ;RESET 'PARAMETERS LOADED' INDICATOR
020352  012605                           MOV     (SP)+,R5         ;RESTORE R5
020354  000207                           RTS     PC               ;RETURN

                                ;ROUTINE TO COMPRESS A LIST
                                ;CALL:
                                ;        MOV     #ADDRS,R1        ;COMPRESS LIST STARTING AT THIS ADDRESS
                                ;        JSR     PC,CMPRES
                                ;        RETURN

020356  016111  000002          CMPRES:  MOV     2(R1),(R1)       ;COMPRESS THE TABLE IN R1
020362  001403                           BEQ     1$               ;BR WHEN ZERO FOUND
020364  062701  000002                   ADD     #2,R1            ;INCREMENT R1
020370  000772                           BR      CMPRES           ;CONTINUE COMPRESSING TABLE
020372  000207                  1$:      RTS     PC               ;RETURN

                                ;ROUTINE TO SETUP PARAMETERS FOR A SEQUENTIAL READ OR WRITE OF THE DISK
                                ;CALL:
                                ;        MOV     #DPB,R0          ;DPB ADDRESS
                                ;        MOV     #-1,$PACK(R0)    ;'WRITE PACK' FLAG
                                ;  OR
                                ;        MOV     #1,$PACK(R0)     ;'READ PACK' FLAG
                                ;        JSR     PC,WRTPK
                                ;        RETURN

020374  004737  035014          WRTPK:   JSR     PC,$RAND         ;CYCLE THE RANDOM NUMBER GENERATOR
020400  005760  000040                   TST     $OPERC+2(R0)     ;SEE IF FIRST OPERATION
020404  001007                           BNE     WRTPK1           ;BR IF UPPER WORD OF COUNTER NOT ZERO
020406  005760  000036                   TST     $OPERC(R0)       ;LOWER WORD ZERO ?
020412  001004                           BNE     WRTPK1           ;BR IF NOT 1ST OPERATION
020414  105760  000026                   TSTB    $PACK(R0)        ;SEE WHICH - 'R' OR 'W'
020420  100530                           BMI     WRTPK3           ;BR IF 'W'
020422  000515                           BR      WRTPK2           ;'R' OPERATION
020424  116060  002116  000027  WRTPK1:  MOVB    $RMCS1(R0),$PREV0(R0)   ;SAVE CURRENT PARAMETERS
020432  004737  023176                   JSR     PC,READDR        ;GET THE DECREMENTED SECTOR AND TRACK ADDRESSES
020436  012660  000034                   MOV     (SP)+,$PREVA+2(R0)       ;CYLINDER ADDRESS
020442  112660  000033                   MOVB    (SP)+,$PREVA+1(R0)   ;TRACK ADDRESS
020446  112660  000032                   MOVB    (SP)+,$PREVA(R0)   ;SECTOR ADDRESS
020452  016060  002124  000010           MOV     $RMDA(R0),$SEC(R0)   ;NEW SECTOR & TRACK ADDRESS
020460  016060  002152  000012           MOV     $RMDC(R0),$CYL(R0)   ;NEW CYLINDER ADDRESS
020466  026060  000012  000106           CMP     $CYL(R0),MAXCYL(R0)   ;SEE IF AT END
020474  103436                           BLO     2$               ;BR IF LESS THAN 'MAXCYL'
020476  101004                           BHI     1$               ;BR IF GREATER THAN 'MAXCYL'
020500  126060  000011  000112           CMPB    $TRK(R0),MAXTRK(R0)   ;SEE IF AT MAX TRACK
020506  103431                           BLO     2$               ;BR IF NOT GREATER
020510  116060  000114  000011  1$:      MOVB    MINTRK(R0),$TRK(R0)   ;RESET TRACK ADDRESS
020516  116060  000120  000010           MOVB    MINSEC(R0),$SEC(R0)   ;RESET SECTOR ADDRESS
020524  016060  000110  000012           MOV     MINCYL(R0),$CYL(R0)   ;RESET CYLINDER ADDRESS
020532  112760  000004  000024           MOVB    #4,$CODE(R0)     ;SET CODE TO READ DATA
```

```
020540  122760  177776  000026            CMPB    #-2,$PACK(R0)      ;WT OPERATION IN PROCESSING
020546  001475                            BEQ     WRTPK5             ;YES
020550  004737  030254                    JSR     PC,EOP2            ;DROP THE DRIVE (NORMAL TERMINATION)

020554  032777  000020  160372            BIT     #SW04,@SWR         ;IS SWITCH 4 SET ?
020562  001003                            BNE     2$                 ;BR IF SET
020564  005726                            TST     (SP)+              ;INCREMENT THE STACK POINTER
020566  000137  006100                    JMP     MAIN               ;RETURN DIRECTLY TO 'MAIN'
020572  013760  001466  000020  2$:       MOV     MAXDL,$WRDL(R0)     ;BUFFER SIZE IS MAXIMUM
020600  042760  000377  000020            BIC     #377,$WRDL(R0)     ;SECTOR BOUNDRY FOR WRITTING
020606  013760  001466  000004            MOV     MAXDL,$WRDM(R0)     ;WORD COUNT
020614  042760  000377  000004            BIC     #377,$WRDM(R0)     ;SECTOR BOUNDRY FOR WRITTING
020622  005760  000004                    TST     $WRDM(R0)          ;SIZE=0 ?
020626  003006                            BGT     3$                 ;NO
020630  012760  000400  000004            MOV     #256.,$WRDM(R0)    ;SET TO ONE SECTOR
020636  012760  000400  000020            MOV     #256.,$WRDL(R0)    ;SET ONE SECTOR
020644  005460  000004          3$:       NEG     $WRDM(R0)          ;CHANGE WORD COUNT TO 2'S COMPLEMENT
020650  105760  000026                    TSTB    $PACK(R0)          ;READ OR WRITE ?
020654  100412                            BMI     WRTPK3             ;BR IF WRITE
020656  012760  000402  000022  WRTPK2:   MOV     #258.,$SSEC(R0)     ;SECTOR SIZE FOR READ
020664  112760  000005  000024            MOVB    #5,$CODE(R0)       ;CODE FOR READ HEADER & DATA
020672  112760  000173  000002            MOVB    #RDHD,$COMND(R0)     ;DRIVE CODE FOR OPERATION
020700  000415                            BR      WRTPK4             ;SET UP FOR EXIT
020702  012760  000400  000022  WRTPK3:   MOV     #256.,$SSEC(R0)    ;SECTOR SIZE
020710  112760  000002  000024            MOVB    #2,$CODE(R0)       ;CODE FOR WRTDAT
020716  112760  000161  000002            MOVB    #WRTDAT,$COMND(R0)  ;OP CODE
020724  004737  020022                    JSR     PC,GETPAT          ;GET PATTERN CODE
020730  110560  000030                    MOVB    R5,$PATTC(R0)      ;PATTERN CODE
020734  005060  000104          WRTPK4:   CLR     $NEXT(R0)          ;CLEAR 'PARAMETER SELECTED' INDICATOR
020740  000207                            RTS     PC                 ;RETURN
020742  005037  001320          WRTPK5:   CLR     PACK               ;CLEAR WT FLAG
020746  105060  000026                    CLRB    $PACK(R0)          ;CLEAR WT FLAG
020752  005726                            TST     (SP)+              ;CLEAR STACK LEVEL
020754  000137  006100                    JMP     MAIN               ;JUMP TO MAIN BACKGROUND LOOP

                                ;ROUTINE TO DETERMINE OF ERROR IS AT A LOCATION ON THE PACK DEFINED
                                ;         IN THE BAD TRACK/SECTOR TABLE FOR THE DRIVE.
                                ;CALL:
                                ;         JSR     PC,SPOTCK
                                ;         RETURN1                    ;ERROR AT AN ADDRESS IN TABLE
                                ;         RETURN2                    ;NO TABLE ENTRY FOR ERROR ADDRESS OR
                                ;                                    ;PARAMETER 'NOTPRT' IS 0

020760                          SPOTCK:
020760  010146                            MOV     R1,-(SP)           ;;PUSH R1 ON STACK
020762  012701  000124                    MOV     #$BDSEC,R1         ;INCREMENT FOR BAD SECTOR TABLE
020766  060001                            ADD     R0,R1              ;ADD THE BLOCK'S STARTING ADDRESS
020770  004737  023176          1$:       JSR     PC,READDR          ;DECREMENT THE SECTOR/TRACK ADDRESS
020774  021126                            CMP     (R1),(SP)+         ;ON THE SAME CYLINDER ?
020776  001023                            BNE     5$                 ;BRANCH IF NOT
021000  122761  177777  000003            CMPB    #-1,3(R1)          ;ALL BAD TRACKS ?
021006  001002                            BNE     2$                 ;BR IF NO
021010  005726                            TST     (SP)+              ;ADJUST STACK AND
021012  000403                            BR      3$                 ;GO CHECK SECTORS
021014  122661  000003          2$:       CMPB    (SP)+,3(R1)        ;COMPARE THE TRACK ADDRESS
021020  001013                            BNE     6$                 ;BR IF IT IS NOT EQUAL
021022  122761  177777  000002  3$:       CMPB    #-1,2(R1)          ;ALL BAD SECTORS ?
```

```
021030  001002                                  BNE     4$              ;BR IF NO
021032  005726                                  TST     (SP)+           ;ADJUST STACK AND
021034  000413                                  BR      8$              ;CHECK 'NOTPRT'
021036  122661  000002                  4$:     CMPB    (SP)+,2(R1)     ;COMPARE THE SECTOR ADDRESS
021042  001003                                  BNE     7$              ;BR IF NOT EQUAL
021044  000407                                  BR      8$              ;CHECK 'NOTPRT'
021046  005726                          5$:     TST     (SP)+           ;CLEAR OFF THE STACK
021050  005726                          6$:     TST     (SP)+           ;INCREMENT THE STACK POINTER
021052  062701  000004                  7$:     ADD     #4,R1           ;GO TO THE NEXT LOCATION IN THE TABLE
021056  005711                                  TST     (R1)            ;EMPTY ENTRY OR TERMINATOR ?
021060  100407                                  BMI     9$              ;BR IF YES
021062  000742                                  BR      1$              ;TRY NEXT SECTOR
021064  005737  001510                  8$:     TST     NOTPRT          ;PRINT THE ERROR ANYWAY ?
021070  001006                                  BNE     10$             ;BR IF NOT
021072  012737  177777  001364                  MOV     #-1,BADSEC      ;SET THE INDICATOR FOR THE IDENTIFICATION LINE
021100  062766  000002  000002          9$:     ADD     #2,2(SP)        ;INCREMENT THE RETURN
021106                                  10$:
021106  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
021110  000207                                  RTS     PC              ;RETURN


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

                                        .SBTTL  ERROR MESSAGE GENERATION ROUTINES

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

                                        ;PRINT LINE 1 OF ERROR MESSAGE:
                                        ;'HH:MM:SS'

021112  032777  002000  160034          LINE1:  BIT     #SW10,@SWR      ;SWITCH 10 SET ?
021120  001402                                  BEQ     1$              ;BR IF NOT
021122  104401  001176                          TYPE    ,$BELL          ;RING THE BELL
021126  032777  020000  160020          1$:     BIT     #SW13,@SWR      ;INHIBIT TYPEOUT ?
021134  001405                                  BEQ     2$              ;BR IF NOT
021136  104414  001203                          DISPLY  ,$CRLF          ;CR-LF
021142  104414  001203                          DISPLY  ,$CRLF          ;CR-LF
021146  000406                                  BR      3$              ;EXIT
021150  104414  001203                  2$:     DISPLY  ,$CRLF          ;CR-LF
021154  004737  024364                          JSR     PC,$TIME        ;TYPE THE TIME
021160  104414  073271                          DISPLY  ,BLNKS1         ;TYPE 1 BLANK
021164  000207                          3$:     RTS     PC              ;RETURN & TYPE DESCRIPTION

                                        ;PRINT LINE 2 OF ERROR MESSAGE
                                        ;'PRESENT ORDER = XXXX  PREVIOUS ORDER = XXXX'
                                        ;'* ERROR AT BAD TRACK/SECTOR'
                                        ;'DRV RMCS1   RMCS2   RMDS1   RMER1   RMMR2   RMER2   RMEC1   RMEC2'
                                        ;'RMWC    RMBA    RMDA    RMAS    RMLA    RMDB    RMMR1   RMDT'
                                        ;'RMSN    RMOF    RMDC    RMCC    STATUS'
                                        ;'BUS ADDRESS OR WORD COUNT NOT CONSISTENT'
                                        ;'RMBA = XXXXXX   RMWC = XXXXXX'
                                        ;'BUFFER ADR = XXXXXX   SIZE = XXXX   ACTUAL NMBR WRDS XFRD = XXX'

021166                                  LINE2:
021166  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
021170  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
021172  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
021174  104414  001203                          DISPLY  ,$CRLF          ;CR-LF
```

```
021200  005037  021326                    CLR     4$                  ;CLEAR MESSAGE ADDRESS STORAGE
021204  005004                            CLR     R4                  ;WORKING REGISTER
021206  012737  071374  021326            MOV     #LIN2C,4$           ;ADDRESS OF 'PRESENT ORDER = ' MSG
021214  116004  002116                    MOVB    $RMCS1(R0),R4       ;GET THE OPCODE
021220  042704  177701                    BIC     #^C76,R4            ;SAVE ONLY SIGNIFICANT BITS
021224  004737  021262                    JSR     PC,1$               ;TYPE THE FIRST MNEMONIC
021230  005737  021332                    TST     5$                  ;SEE IF MNEMONIC ENTRY FOUND
021234  001440                            BEQ     LINE2A              ;BR IF NOT
021236  012737  071415  021326            MOV     #LIN2P,4$           ;ADDRESS OF 'PREVIOUS ORDER = ' MSG
021244  116004  000027                    MOVB    $PREVO(R0),R4       ;PREVIOUS OPERATION CODE
021250  042704  177701                    BIC     #^C76,R4            ;SAVE ONLY SIGNIFICANT BITS
021254  004737  021262                    JSR     PC,1$               ;TYPE THE PREVIOUS MNEMONIC
021260  000426                            BR      LINE2A              ;CONTINUE
021262  005005                    1$:     CLR     R5                  ;CLEAR THE TABLE INDEX
021264  126504  002134            2$:     CMPB    OPTBL(R5),R4        ;LOOK FOR THE OPCODE
021270  001405                            BEQ     3$                  ;BR WHEN OPCODE COUNT EQUALS OPCODE
021272  105765  002134                    TSTB    OPTBL(R5)           ;LOOK FOR END OF TABLE
021276  100402                            BMI     3$                  ;BR IF END
021300  005205                            INC     R5                  ;INCREMENT THE POINTER
021302  000770                            BR      2$                  ;CONTINUE - NOT END OF TABLE
021304  006305                    3$:     ASL     R5                  ;SHIFT INDEX
021306  006305                            ASL     R5                  ;SHIFT THE INDEX
021310  006305                            ASL     R5                  ;SHIFT THE INDEX
021312  012737  002156  021332            MOV     #MNTBL,5$           ;ADDRESS OF ASCII TEXT TABLE
021320  060537  021332                    ADD     R5,5$               ;ADD THE INDEX
021324  104414                            DISPLY                      ;TYPE IT
021326  000000                    4$:     .WORD   0                   ;ADDRESS OF 'PRESENT' OR 'PREVIOUS' MESSAGE
021330  104414                            DISPLY                      ;TYPE THE OPERATION MNEMONIC
021332  000000                    5$:     .WORD   0                   ;ADDRESS OF MESSAGE
021334  000207                            RTS     PC                  ;RETURN TO MAIN ROUTINE
021336  005737  001364            LINE2A: TST     BADSEC              ;PRINT THE BAD SECTOR LINE ?
021342  001404                            BEQ     LINE2B              ;BR IF NOT
021344  104414  001203                    DISPLY  ,$CRLF              ;CR-LF
021350  104414  071441                    DISPLY  ,LIN2S              ;ERROR ADDRESS DEFINED AS BAD AREA
021354  104414  001203            LINE2B: DISPLY  ,$CRLF              ;CR-LF
021360  104414  070770                    DISPLY  ,DH14               ;STANDARD RM REGISTER HEADER
021364  104414  073271                    DISPLY  ,BLNKS1             ;TYPE 1 BLANK
021370  013746  001346                    MOV     UNIT,-(SP)          ;PUT THE DRIVE NUMBER ON THE STACK
021374  004737  023156                    JSR     PC,LINDEC           ;TYPE DRIVE NUMBER
021400  104414  073270                    DISPLY  ,BLNKS2             ;TYPE 2 BLANKS
021404  012705  071314                    MOV     #DT14,R5            ;REGISTER INDEXES
021410  004737  021540                    JSR     PC,3$               ;PRINT THE REGISTERS
021414  032777  000040  157532            BIT     #SW05,@SWR          ;PRINT THE OPTIONAL REGISTERS ?
021422  001014                            BNE     1$                  ;BR IF NOT
021424  104414  071073                    DISPLY  ,DH15
021430  012705  071336                    MOV     #DT15,R5            ;SECOND DATA LINE
021434  004737  021540                    JSR     PC,3$               ;PRINT THEM
021440  104414  071172                    DISPLY  ,DH16
021444  012705  071360                    MOV     #DT16,R5            ;THIRD DATA LINE
021450  004737  021540                    JSR     PC,3$               ;PRINT THE REGISTERS
021454  032760  000100  000016    1$:     BIT     #BIT6,$STATUS(R0)   ;DATA ERROR ?
021462  001422                            BEQ     2$                  ;BR IF NOT
021464  016046  000020                    MOV     $WRDL(R0),-(SP)     ;TRANSFER SIZE
021470  066016  002120                    ADD     $RMWC(R0),(SP)      ;ADD REMAINING WORD COUNT
021474  006316                            ASL     (SP)                ;CONVERT TO AN BYTE INCREMENT
021476  066016  000006                    ADD     $BUF(R0),(SP)       ;BUFFER STARTING ADDRESS
021502  022660  002122                    CMP     (SP)+,$RMBA(R0)     ;CORRECT BUFFER ADDRESS ?
```

```
021506 001410                           BEQ     2$                ;BR IF YES
021510 104414  070415                   DISPLY  ,EM46             ;'BUS ADDRESS AND WORD COUNT ARE NOT CONSISTENT'
021514 104414  001203                   DISPLY  ,$CRLF            ;CR-LF
021520 004737  021632                   JSR     PC,LINE3D         ;PRINT LINE 3D OF ERROR MESSAGE
021524 004737  022244                   JSR     PC,LINE4          ;PRINT LINE 4 OF ERROR MESSAGE
021530                          2$:
021530 012605                           MOV     (SP)+,R5          ;;POP STACK INTO R5
021532 012604                           MOV     (SP)+,R4          ;;POP STACK INTO R4
021534 012603                           MOV     (SP)+,R3          ;;POP STACK INTO R3
021536 000207                           RTS     PC                ;RETURN TO ERROR PROCESSING ROUTINE
021540 012546                   3$:     MOV     (R5)+,-(SP)       ;PUT THE REGISTER INDEX ON THE STACK
021542 060016                           ADD     R0,(SP)           ;ADD DRIVE'S TABLE ADDRESS
021544 017646  000000                   MOV     @(SP),-(SP)       ;VALUE
021550 004737  023124                   JSR     PC,LINOCT         ;TYPE IT
021554 005726                           TST     (SP)+             ;CORRECT THE STACK POINTER
021556 104414  073270                   DISPLY  ,BLNKS2           ;TYPE 2 BLANKS
021562 005715                           TST     (R5)              ;AT END OF LINE ?
021564 001365                           BNE     3$                ;BR IF NOT
021566 104414  001203           4$:     DISPLY  ,$CRLF            ;CR-LF
021572 000207                           RTS     PC                ;RETURN

                                ;PRINT LINE 3 OF ERROR MESSAGE
                                ;'ERROR AT CCC TT SS   PREVIOUS ADR = CCC TT SS'

021574 104414  071475           LINE3:  DISPLY  ,LINM3            ;LINE 3 ENTRANCE
021600 000517                           BR      LIN3.1            ;FINISH PRINTOUT

                                ;PRINT LINE 3A OF ERROR MESSAGE
                                ;'START CYL = CCC   END CYL = CCC'

021602 104414  071513           LINE3A: DISPLY  ,LINN3            ;LINE 3A ENTRANCE
021606 000514                           BR      LIN3.1            ;FINISH ERROR LINE

                                ;PRINT LINE 3B OF ERROR MESSAGE
                                ;'START CYL = CCC   END CYL = CCC   ACTUAL CYL = CCC'

021610 004737  022146           LINE3B: JSR     PC,LIN3.3         ;LINE 3B ENTRANCE
021614 104414  001203                   DISPLY  ,$CRLF
021620 000207                           RTS     PC

                                ;PRINT LINE 3C OF ERROR MESSAGE
                                ;'START CYL = CCC   END CYL = CCC   ACTUAL CYL = CCC   TRK = TT'

021622 004737  022146           LINE3C: JSR     PC,LIN3.3         ;LINE 3C ENTRANCE
021626 000137  022200                   JMP     LIN3.4            ;FINISH MESSAGE

                                ;PRINT LINE 3D OF ERROR MESSAGE
                                ;'RMBA = XXXXXX   RMWC = XXXXXX'

021632 032777  000040  157314   LINE3D: BIT     #SW05,@SWR        ;SWITCH 5 SET ?
021640 001416                           BEQ     1$                ;BR IF IT IS
021642 104414  071664                   DISPLY  ,LINB3            ;'RMBA = '
021646 016046  002122                   MOV     $RMBA(R0),-(SP)   ;BUFFER ADDR REG CONTENTS
021652 004737  023124                   JSR     PC,LINOCT         ;CONVERT TO OCTAL AND TYPE IT
021656 104414  071674                   DISPLY  ,LINW3            ;'  RMWC = '
021662 016046  002120                   MOV     $RMWC(R0),-(SP)   ;WORD COUNT REGISTER CONTENTS
021666 004737  023124                   JSR     PC,LINOCT         ;CONVERT TO OCTAL AND TYPE IT
```

```
021672  104414  001203                          DISPLY  ,$CRLF
021676  000207                          1$:     RTS     PC


                                        ;PRINT LINE 3E OF ERROR MESSAGE
                                        ;'START CYL = CCC   START TRK = TT   START SEC = SS'

021700  104414  071560                  LINE3E: DISPLY  ,LINS3             ;'START CYL = '
021704  016046  000012                          MOV     $CYL(R0),-(SP)     ;MOVE CYL TO STACK
021710  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
021714  104414  073270                          DISPLY  ,BLNKS2            ;TYPE 2 BLANKS
021720  104414  071706                          DISPLY  ,LINST3            ;'START TRK = '
021724  005046                                  CLR     -(SP)              ;CLEAR STACK
021726  116016  000011                          MOVB    $TRK(R0),(SP)      ;TRACK TO STACK
021732  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
021736  104414  073270                          DISPLY  ,BLNKS2            ;TYPE 2 BLANKS
021742  104414  071723                          DISPLY  ,LINSS3            ;'START SEC = '
021746  005046                                  CLR     -(SP)              ;CLEAR STACK
021750  116016  000010                          MOVB    $SEC(R0),(SP)      ;SECTOR ADDR TO STACK
021754  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
021760  104414  001203                          DISPLY  ,$CRLF
021764  000207                                  RTS     PC


                                        ;PRINT LINE 3F OF ERROR MESSAGE
                                        ;'RMDA = XXXXXX   RMCA = XXXXXX'

021766  032777  000040  157160          LINE3F: BIT     #SW5,@SWR          ;SWITCH 5 SET ?
021774  001420                                  BEQ     1$                 ;BR IF NOT
021776  104414  071654                          DISPLY  ,LINDA3            ;'RMDA = '
022002  016046  002124                          MOV     $RMDA(R0),-(SP) ;PUT SECTOR/TRACK ADDRESS ON THE STACK
022006  004737  023124                          JSR     PC,LINOCT          ;TYPE IT
022012  104414  073270                          DISPLY  ,BLNKS2            ;TYPE 2 BLANKS
022016  104414  071643                          DISPLY  ,LINCA3            ;' RMDC = '
022022  016046  002152                          MOV     $RMDC(R0),-(SP) ;PUT DESIRED CYLINDER ADDRESS ON THE STACK
022026  004737  023124                          JSR     PC,LINOCT          ;TYPE IT
022032  104414  001203                          DISPLY  ,$CRLF
022036  000207                          1$:     RTS     PC


                                        ;'CCC TT SS   PREV ADR = CCC TT SS'

022040  004737  023176                  LIN3.1: JSR     PC,READDR          ;DECREMENT TRACK AND SECTOR ADDRESS
022044  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
022050  104414  071510                          DISPLY  ,T                 ;PRINT ' T'
022054  004737  023156                          JSR     PC,LINDEC          ;TYPE TRACK IN DECIMAL
022060  104414  071534                          DISPLY  ,S                 ;PRINT ' S'
022064  004737  023156                          JSR     PC,LINDEC          ;TYPE SECTOR ADDRESS
022070  104414  071537                          DISPLY  ,LINP3             ;PRINT 'PREV ADDR'
022074  016046  000034                          MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
022100  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
022104  104414  071510                          DISPLY  ,T                 ;PRINT ' T'
022110  005046                                  CLR     -(SP)              ;MAKE ROOM ON THE STACK
022112  116016  000033                          MOVB    $PREVA+1(R0),(SP)   ;PREVIOUS TRACK ADDRESS
022116  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
022122  104414  071534                          DISPLY  ,S                 ;PRINT ' S'
022126  005046                                  CLR     -(SP)              ;MAKE ROOM ON THE STACK
022130  116016  000032                          MOVB    $PREVA(R0),(SP)    ;PREVIOUS SECTOR DDRESS
022134  004737  023156                          JSR     PC,LINDEC          ;TYPE IT IN DECIMAL
022140  104414  001203                          DISPLY  ,$CRLF
```

```
022144  000207                                  RTS     PC

                                        ;'START CYL = CCC   END CYL = CCC'

022146  104414  071560                  LIN3.3: DISPLY  ,LINS3              ;LINE '3B & 3C' ENTRANCE
022152  016046  000034                          MOV     $PREVA+2(R0),-(SP)  ;PREVIOUS CYLINDER
022156  004737  023156                          JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
022162  104414  071575                          DISPLY  ,LINEN3             ;PRINT 'END CYL'
022166  016046  002152                          MOV     $RMDC(R0),-(SP)     ;PRESENT CYLINDER
022172  004737  023156                          JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
022176  000207                                  RTS     PC

                                        ;'ACTUAL CYL = CCC   TRK = TT'

022200  104414  071612                  LIN3.4: DISPLY  ,LINA3              ;PRINT 'ACTUAL'
022204  013746  076746                          MOV     CYLNDR,-(SP)        ;ACTUAL CYLINDER
022210  042716  010000                          BIC     #BIT12,(SP)         ;CLEAR THE FORMAT BIT
022214  004737  023156                          JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
022220  104414  071632                          DISPLY  ,LINT3              ;PRINT TRACK
022224  005046                                  CLR     -(SP)               ;CLEAR STACK WORD
022226  116016  002125                          MOVB    $RMDA+1(R0),(SP)    ;PUT TRACK ON STACK
022232  004737  023156                          JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
022236  104414  001203                          DISPLY  ,$CRLF
022242  000207                                  RTS     PC

                                        ;PRINT LINE 4 OF ERROR MESSAGE
                                        ;'BUFFER ADR = XXXXXX   SIZE = XXXX   ACTUAL NMBR WRDS XFRD = XXX'

022244  032760  000100  000016          LINE4:  BIT     #BIT06,$STATUS(R0)  ;DATA ERROR ?
022252  001427                                  BEQ     1$                  ;BR IF NOT
022254  104414  071740                          DISPLY  ,LINM4              ;'PRINT BUFFER'
022260  016046  000006                          MOV     $BUF(R0),-(SP)      ;BUFFER ADDR ON STACK
022264  004737  023124                          JSR     PC,LINOCT           ;CONVERT TO OCTAL & PRINT
022270  104414  071757                          DISPLY  ,LINS4              ;PRINT 'SIZE'
022274  016046  000020                          MOV     $WRDL(R0),-(SP)     ;BUFFER SIZE
022300  004737  023156                          JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
022304  104414  071771                          DISPLY  ,LINX4              ;'ACTUAL NMBR WRDS XFRD = '
022310  016046  002122                          MOV     $RMBA(R0),-(SP)     ;VALUE IN BUFFER ADDR REGISTER
022314  166016  000006                          SUB     $BUF(R0),(SP)       ;SUBTRACT STARTING ADDRESS
022320  006216                                  ASR     (SP)                ;CONVERT INTO A WORD COUNT
022322  004737  023156                          JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
022326  104414  001203                          DISPLY  ,$CRLF              ;CR-LF
022332  000207                          1$:     RTS     PC                  ;RETURN

                                        ;PRINT LINE 5 OF ERROR MESSAGE
                                        ;'GOOD DATA = XXXXXX   BAD DATA = XXXXXX   SECT POS = XXX'

022334  104414  072024                  LINE5:  DISPLY  ,LIND5              ;PRINT 'GOOD DATA'
022340  162760  000002  002122                  SUB     #2,$RMBA(R0)        ;BACK THE ADDRESS UP
022346  017046  002122                          MOV     @$RMBA(R0),-(SP)    ;'GOOD' DATA - AT THE BUFFER LOCATION
022352  004737  023124                          JSR     PC,LINOCT           ;TYPE IT
022356  104414  072041                          DISPLY  ,LINB5              ;PRINT 'BAD DATA'
022362  016046  002140                          MOV     $RMDB(R0),-(SP)     ;BAD DATA FROM BUFFER
022366  004737  023124                          JSR     PC,LINOCT           ;TYPE IT
022372  016046  002120                          MOV     $RMWC(R0),-(SP)     ;WORD LENGTH ON STACK
022376  066016  000020                          ADD     $WRDL(R0),(SP)      ;MAKE INTO A POSITIVE NUMBER
022402  005046                                  CLR     -(SP)               ;UPPER DIVIDEND TO ZERO
```

```
022404  016046  000022                  MOV     $SSEC(R0),-(SP)  ;SECTOR SIZE ON THE STACK
022410  004737  030610                  JSR     PC,LINKDV        ;DIVIDE WORDS XFERED BY SECTOR SIZE
022414  012616                          MOV     (SP)+,(SP)       ;MOVE REMAINDER UP THE STACK
022416  104414  072057                  DISPLY  ,LINP5           ;PRINT 'SECT POS'
022422  004737  023156                  JSR     PC,LINDEC        ;TYPE THE POSITION
022426  104414  001203                  DISPLY  ,$CRLF
022432  000207                          RTS     PC


                                ;PRINT LINE 5A OF THE ERROR MESSAGE
                                ;'HEADER FROM ERROR SECTOR  XXXXXX  XXXXXX  XXXXXX  XXXXXX'

022434                          LINE5A:
022434  104414  072075          2$:     DISPLY  ,LINS5           ;'HEADER CONTENTS OF ERROR SECTOR'
022440  013746  076746                  MOV     CYLNDR,-(SP)     ;HEADER POSITION
022444  004737  023124                  JSR     PC,LINOCT        ;TYPE IT
022450  104414  073270                  DISPLY  ,BLNKS2          ;TYPE 2 BLANKS
022454  013746  076750                  MOV     CYLNDR+2,-(SP)   ;HEADER POSITION +2
022460  004737  023124                  JSR     PC,LINOCT        ;TYPE IT
022464  104414  073270                  DISPLY  ,BLNKS2          ;TYPE 2 BLANKS
022470  104414  073273                  DISPLY  ,LINX5           ;APPENDING INFO 1/23/77
022474  104414  001203          3$:     DISPLY  ,$CRLF
022500  000207                          RTS     PC


                                ;PRINT LINE 5B OF ERROR MESSAGE
                                ;'RMEC1 = XXXXXX   RMEC2 = XXXXXX'

022502  104414  072131          LINE5B: DISPLY  ,LINEP5          ;'RMEC1 = '
022506  016046  002162                  MOV     $RMEC1(R0),-(SP)  ;PUT REGISTER CONTENTS ON THE STACK
022512  004737  023124                  JSR     PC,LINOCT        ;TYPE IT
022516  104414  073270                  DISPLY  ,BLNKS2          ;TYPE 2 BLANKS
022522  104414  072142                  DISPLY  ,LINEO5          ;' RMEC2 = '
022526  016046  002164                  MOV     $RMEC2(R0),-(SP)  ;PUT REGISTER CONTENTS ON THE STACK
022532  004737  023124                  JSR     PC,LINOCT        ;TYPE IT
022536  104414  001203                  DISPLY  ,$CRLF
022542  000207                          RTS     PC               ;RETURN


                                ;PRINT LINE 6 OF ERROR MESSAGE
                                ;'SECTOR IS ECC CORRECTABLE'

022544  104414  072154          LINE6:  DISPLY  ,LINB6           ;ECC CORRECTABLE
022550  104414  001203                  DISPLY  ,$CRLF
022554  000207                          RTS     PC


                                ;PRINT LINE 6A OF THE ERROR MESSAGE
                                ;'SECTOR READ CORRECTLY AT OFFSET N'

022556  104414  072207          LINE6A: DISPLY  ,LINC6           ;PRINT 'READ CORRECTLY AT OFFSET N'
022562  000411                          BR      LIN6.1           ;TYPE THE REST OF THE LINE

                                ;PRINT LINE 6B OF THE ERROR MESSAGE
                                ;'SECTOR IS ECC CORRECTABLE AT OFFSET N'

022564  104414  072154          LINE6B: DISPLY  ,LINB6           ;PRINT 'SECTOR IS ECC CORRECTABLE '
022570  000406                          BR      LIN6.1

                                ;PRINT LINE 6C OF THE ERROR MESSAGE
                                ;'CORRECTED ON NTH RETRY'
```

```

022572  104414  072236                  LINE6C: DISPLY  ,LING6            ;'CORRECTED ON NTH RETRY'
022576  000414                                  BR      LIN6.2            ;TYPE THE REST OF THE LINE

                                        ;PRINT LINE 6D OF THE ERROR MESSAGE
                                        ;'UNCORRECTABLE AFTER N RETRIES'

022600  104414  072265                  LINE6D: DISPLY  ,LINU06           ;'UNCORRECTABLE AFTER N RETRIES'
022604  000411                                  BR      LIN6.2            ;FINISH

                                        ;TYPE THE OFFSET VALUE IN MICRO-INCHES

022606  006301                          LIN6.1: ASL     R1                ;DOUBLE THE OFFSET TABLE INDEX
022610  016137  002474  022620                  MOV     OFMTBL(R1),1$     ;ADDRESS OF OFFSET POSITION MESSAGE
022616  104414                                  DISPLY
022620  000000                          1$:     .WORD   0                 ;OFFSET VALUE
022622  104414  001203                          DISPLY  ,$CRLF
022626  000207                                  RTS     PC

                                        ;RETRY COUNT TYPEOUT

022630  005046                          LIN6.2: CLR     -(SP)             ;CLEAR STACK
022632  113716  001353                          MOVB    RETRY+1,(SP)      ;RETRY COUNT
022636  004737  023156                          JSR     PC,LINDEC         ;TYPE IT IN DECIMAL
022642  104414  072254                          DISPLY  ,LINR6            ;'RETRY'
022646  104414  001203                          DISPLY  ,$CRLF
022652  000207                                  RTS     PC

                                        ;PRINT LINE 7 OF THE ERROR MESSAGE
                                        ;'ORDERS:XXXXX   TOTAL ERRORS:XXX   WRDS XFRD:XXXXXXX   WRDS READ:XXXXXXX'

022654  104414  072340                  LINE7:  DISPLY  ,LIN7O            ;PRINT ORDER COUNT
022660  012746  000036                          MOV     #$OPERC,-(SP)     ;TO STACK
022664  060016                                  ADD     R0,(SP)           ;ADD THE BASE ADDRESS
022666  004737  035212                          JSR     PC,$DB2D          ;CONVERT IT
022672  004737  031124                          JSR     PC,$SUPRS         ;PRINT IT
022676  104414  072412                          DISPLY  ,LIN7T            ;TOTAL ERRORS
022702  016046  000056                          MOV     $TOTAL(R0),-(SP)          ;TO STACK
022706  004737  023156                          JSR     PC,LINDEC         ;TYPE IT IN DECIMAL
022712  104414  072424                          DISPLY  ,LIN7X            ;PRINT 'WRDS XFR'
022716  012746  000046                          MOV     #$TRANS,-(SP)     ;ADDRESS OF LOW WORD ON STACK
022722  060016                                  ADD     R0,(SP)
022724  004737  035212                          JSR     PC,$DB2D          ;CONVERT
022730  004737  031124                          JSR     PC,$SUPRS         ;PRINT
022734  104414  072440                          DISPLY  ,LIN7R            ;'BITS READ'
022740  012746  000052                          MOV     #$READ,-(SP)      ;LOW WORD ADDRESS
022744  060016                                  ADD     R0,(SP)
022746  004737  035212                          JSR     PC,$DB2D          ;CONVERT
022752  004737  031124                          JSR     PC,$SUPRS         ;PRINT IT
022756  104414  001203                          DISPLY  ,$CRLF            ;CR-LF
022762  032777  100000  156164                  BIT     #SW15,@SWR        ;SEE IF 'HALT ON ERROR' - SWITCH 15
022770  001401                                  BEQ     1$                ;BR IF NOT
022772  000000                                  HALT                      ;SWITCH 15 HALT
022774  000207                          1$:     RTS     PC

                                        ;PRINT LINE 7A OF ERROR MESSAGE
                                        ;'ORDERS:XXXXX   TOTAL SEEKS=XXXXX   TOTAL MISPOS ERR = XXX   TOTAL SKI= XXX'
```

```

 022776  104414  072340          LINE7A: DISPLY  ,LIN7O              ;'ORDERS = '
 023002  012746  000036                  MOV     #$OPERC,-(SP)   ;ORDER COUNT INCREMENT
 023006  060016                          ADD     R0,(SP)             ;ADD BASE ADDRESS
 023010  004737  035212                  JSR     PC,$DB2D            ;CONVERT THE COUNT
 023014  004737  031124                  JSR     PC,$SUPRS           ;PRINT IT
 023020  104414  072350                  DISPLY  ,LIN7P              ;'TOTAL SEEKS = '
 023024  012746  000042                  MOV     #$POSIT,-(SP)       ;TOTAL SEEKS
 023030  060016                          ADD     R0,(SP)             ;DEVICE TABLE ADDRESS
 023032  004737  035212                  JSR     PC,$DB2D            ;CONVERT THE SEEK COUNT
 023036  004737  031124                  JSR     PC,$SUPRS           ;PRINT IT
 023042  104414  072312                  DISPLY  ,LIN7M              ;'  TOTAL MISPOS ERR = '
 023046  016046  000066                  MOV     $MISPO(R0),-(SP)   ;TOTAL ERRORS
 023052  004737  023156                  JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
 023056  104414  072370                  DISPLY  ,LIN7S              ;'  TOTAL SKI,OCYL ERR = '
 023062  016046  000064                  MOV     $SKI(R0),-(SP)      ;CONVERT & PRINT IT
 023066  004737  023156                  JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
 023072  104414  001203                  DISPLY  ,$CRLF              ;CR-LF
 023076  032777  100000  156050          BIT     #SW15,@SWR          ;SEE IF HALT ON ERROR - SWITCH 15 SET
 023104  001401                          BEQ     1$                  ;BR IF NOT
 023106  000000                          HALT                        ;SWITCH 15 HALT
 023110  000207                  1$:     RTS     PC


                                 ;PRINT LINE 8 OF THE ERROR MESSAGE
                                 ;'DIFFERENT ERROR DURING RETRY'

 023112  104414  072455          LINE8:  DISPLY  ,LIN8M
 023116  004737  021166                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
 023122  000207                          RTS     PC

                                 ;OCTAL TYPEOUT ROUTINE
                                 ;CALL:
                                 ;       MOV     NUM,-(SP)           ;PUT THE NUMBER ON THE STACK
                                 ;       JSR     PC,LINOCT
                                 ;       RETURN

 023124  016646  000002          LINOCT: MOV     2(SP),-(SP)         ;PUT NUMBER IN PROPER LOCATION ON STACK
 023130  004737  031554                  JSR     PC,$SB2O            ;CONVERT THE NUMBER TO OCTAL
 023134  012637  023150                  MOV     (SP)+,1$            ;GET THE ADDRESS OF THE ASCII STRING
 023140  062737  000005  023150          ADD     #5.,1$              ;ADDRESS THE LAST 6 ASCII DIGITS
 023146  104414                          DISPLY                      ;TYPE IT
 023150  000000                  1$:     .WORD   0                   ;ADDRESS
 023152  012616                          MOV     (SP)+,(SP)          ;CORRECT THE STACK
 023154  000207                          RTS     PC                  ;RETURN

                                 ;ROUTINE TO CONVERT THE INPUT NUMBER TO DECIMAL AND TYPE IT WITH
                                 ;LEADING ZERO SUPRESSION
                                 ;CALL:
                                 ;       MOV     NUM,-(SP)           ;PUT THE NUMBER ON THE STACK
                                 ;       JSR     PC,LINDEC
                                 ;       RETURN

 023156  016646  000002          LINDEC: MOV     2(SP),-(SP)         ;SET UP STACK FOR CONVERT
 023162  004737  031524                  JSR     PC,$SB2D            ;CONVERT IT TO DECIMAL
 023166  004737  031124                  JSR     PC,$SUPRS           ;TYPE IT (WITH LEADING ZEROS SUPRESSED)
 023172  012616                          MOV     (SP)+,(SP)          ;RESTORE STACK POINTER
 023174  000207                          RTS     PC
```

```

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

                                        .SBTTL  GENERAL SUPPORT SUBROUTINES

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

                                        ;DECREMENT THE SECTOR-TRACK ADDRESS
                                        ;CALL
                                        ;       MOV     #DPB,R0           ;DPB ADDRESS
                                        ;       JSR     PC,READDR
                                        ;       RETURN
                                        ;
                                        ;ON RETURN THE STACK CONTAINS THE FOLLOWING:
                                        ;       4(SP) = SECTOR ADDRESS
                                        ;       2(SP) = TRACK ADDRESS
                                        ;        (SP) = CYLINDER ADDRESS
                                        ;
023176  162706  000006                  READDR: SUB     #6,SP             ;DECREMENT THE STACK POINTER
023202  016616  000006                          MOV     6(SP),(SP)        ;MOVE THE RETURN ADDR DOWN THE STACK
023206  005066  000006                          CLR     6(SP)             ;CLEAR STACK FOR SECTOR
023212  005066  000004                          CLR     4(SP)             ;CLEAR STACK FOR TRACK
023216  116066  002124  000006                  MOVB    $RMDA(R0),6(SP)   ; SECTOR ON STACK
023224  116066  002125  000004                  MOVB    $RMDA+1(R0),4(SP)   ;TRACK ADDRESS
023232  016066  002152  000002                  MOV     $RMDC(R0),2(SP)   ;CYLINDER ADDRESS
023240  005766  000006                          TST     6(SP)             ;SECTOR 0 ?
023244  001403                                  BEQ     1$                ;BRANCH IF SO
023246  105366  000006                          DECB    6(SP)             ;DECREMENT ONE SECTOR
023252  000423                                  BR      4$                ;BRANCH TO EXIT
023254  005766  000004                  1$:     TST     4(SP)             ;ALSO ON TRACK 0 ?
023260  001406                                  BEQ     2$                ;BRANCH IF SO
023262  112766  000037  000006                  MOVB    #31.,6(SP)        ;LAST SECTOR
023270  105366  000004                          DECB    4(SP)             ;DECREMENT ONE TRACK
023274  000412                                  BR      4$                ;EXIT
023276  112766  000037  000006          2$:     MOVB    #31.,6(SP)        ;LAST SECTOR
023304  004737  026546                          JSR     PC,GETLMT         ;GET ADDRESS LIMITS
023310  113766  001450  000004                  MOVB    TRKLMT,4(SP)      ;GET LAST TRACK
023316  005366  000002                          DEC     2(SP)             ;DECREMENT ONE CYLINDER COUNT
023322  000240                          4$:     NOP                       ;DONE
023324  000207                                  RTS     PC                ;RETURN

                                        ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS

023326  012737  177777  001312          CKCLK:  MOV     #-1,CLKFLG        ;CLEAR CLOCK AVAILABILITY FLAG
023334  012737  177777  001310                  MOV     #-1,PCLOCK        ;CLEAR KW11-P CLOCK AVAILABILITY FLAG
023342  012737  023422  000004                  MOV     #CKCLK1,ERRVEC    ;SET UP VECTOR FOR CLOCK CHECK
023350  005037  000006                          CLR     @#ERRVEC+2          ;NEW PSW
023354  005777  155716                          TST     @$LKCSR           ;CHECK FOR KW11-P
023360  005037  001312                          CLR     CLKFLG            ;SET CLOCK AVAILABILITY FLAG
023364  005037  001310                          CLR     PCLOCK            ;SET KW11-P CLOCK FLAG
023370  013701  001302                          MOV     $LPVEC,R1         ;KW11-P VECTOR ADDRESS
023374  012721  024462                          MOV     #CLOCK,(R1)+      ;SET UP KW11-P VECTOR
023400  012711  000300                          MOV     #300,(R1)         ;PSW - PRI 6
023404  012777  174575  155666                  MOV     #-1667.,@$LKCSB   ;LOAD COUNTER BUFFER WITH 16.67
023412  012777  000131  155656                  MOV     #131,@$LKCSR      ;SET CLOCK - CNT UP, 10US, CONT INT
023420  000437                                  BR      CKCLK3
023422  062706  000004                  CKCLK1: ADD     #4,SP             ;RESTORE THE STACK POINTER
```

```
023426  012737  023470  000004          MOV     #CKCLK2,@#ERRVEC  ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
023434  005777  155644                  TST     @$LKS             ;LOOK FOR KW11-L
023440  005037  001312                  CLR     CLKFLG            ;SET CLOCK FLAG
023444  013701  001306                  MOV     $LLVEC,R1         ;KW11-L VECTOR ADDRESS
023450  012721  024462                  MOV     #CLOCK,(R1)+      ;SET UP KW11-L VECTOR
023454  012711  000300                  MOV     #300,(R1)         ;PSW - PRI 6
023460  012777  000100  155616          MOV     #100,@$LKS        ;SET KW11-L INTERRUPT
023466  000414                          BR      CKCLK3
023470  062706  000004          CKCLK2: ADD     #4,SP             ;RESTORE THE STACK POINTER
023474  104401  074173                  TYPE    ,NEDCLK           ;'P OR L CLOCK MUST BE ON SYSTEM'
023500  005737  000042                  TST     42                ;UNDER MONITOR CONTROL ?
023504  001402                          BEQ     1$                ;BR IF NOT
023506  000137  030526                  JMP     $GET42            ;ABORT PROGRAM
023512  000000                  1$:     HALT                      ;HALT
023514  000137  003626                  JMP     START             ;TRY AGAIN
023520  012737  000006  000004  CKCLK3: MOV     #6,@#ERRVEC       ;RESTORE THE ERROR VECTOR
023526  000207                          RTS     PC

                                ;ROUTINE TO DISPLAY STATISTICS FOR ALL DRIVES ASSIGNED
                                ;CALL:
                                ;       JSR     PC,STATPR
                                ;       RETURN

023530  010046                  STATPR: MOV     R0,-(SP)          ;SAVE R0
023532  010446                          MOV     R4,-(SP)          ;SAVE R4
023534  005737  001550                  TST     ASNLST            ;ANY DRIVES ASSIGNED ?
023540  001423                          BEQ     3$                ;BR IF NOT
023542  004737  023644                  JSR     PC,SHDTYP         ;TYPE THE HEADING
023546  005004                          CLR     R4                ;CLEAR THE DRIVE INDEX
023550  006304                  1$:     ASL     R4                ;CHANGE TO INDEX WORDS
023552  016400  002106                  MOV     BLKADR(R4),R0     ;GET THE DRIVE'S BLOCK ADDRESS
023556  006204                          ASR     R4                ;RESTORE R4
023560  136437  035750  001550          BITB    ATABIT(R4),ASNLST  ;IS THIS DRIVE ASSIGNED ?
023566  001404                          BEQ     2$                ;BR IF NOT
023570  004737  023666                  JSR     PC,SDETAL         ;TYPE THE PERFORMANCE SUMMARY
023574  104401  001203                  TYPE    ,$CRLF            ;CR-LF
023600  005204                  2$:     INC     R4                ;INCREMENT THE INDEX
023602  020427  000010                  CMP     R4,#8.            ;FINISHED ?
023606  001360                          BNE     1$                ;BR IF NO
023610  012604                  3$:     MOV     (SP)+,R4          ;RESTORE R4
023612  012600                          MOV     (SP)+,R0          ;RESTORE R0
023614  000207                          RTS     PC                ;RETURN

                                ;ROUTINE TO TYPE THE PERFORMANCE SUMMARY (STATISTICS) FOR AN INDIVIDUAL
                                ;DRIVE.
                                ;CALL:
                                ;       MOV     #DPB,R0           ;DPB ADDRESS
                                ;       JSR     PC,SUMARY
                                ;       RETURN

023616  010046                  SUMARY: MOV     R0,-(SP)          ;SAVE R0
023620  010446                          MOV     R4,-(SP)          ;SAVE R4
023622  004737  023644                  JSR     PC,SHDTYP         ;TYPE THE HEADING
023626  005004                          CLR     R4                ;CLEAR R4 FOR DRIVE NUMBER
023630  111004                          MOVB    (R0),R4           ;DRIVE NUMBER
023632  004737  023666                  JSR     PC,SDETAL         ;TYPE THE STATISTICS
023636  012604                          MOV     (SP)+,R4          ;RESTORE R4
```

```
023640  012600                          MOV     (SP)+,R0                ;RESTORE R0
023642  000207                          RTS     PC                      ;RETURN

                                ;TYPE THE HEADER FOR THE DRIVE PERFORMANCE SUMMARY TYPEOUT
                                ;CALL:
                                ;       JSR     PC,SHDTYP
                                ;       RETURN

023644  004737  024364          SHDTYP: JSR     PC,$TIME                ;TYPE THE TIME OF DAY
023650  004537  031164                  JSR     R5,TYPRI4               ;TYPE AT PRIORITY 4
023654  001203                          $CRLF                           ;CR-LF
023656  004537  031164                  JSR     R5,TYPRI4               ;TYPE THE HEADER
023662  073644                          STATHD                          ;HEADER
023664  000207                          RTS     PC                      ;RETURN

                                ;TYPE THE PERFORMANCE SUMMARY DATE LINE
                                ;CALL:
                                ;       MOV     #DRIVE,R4               ;DRIVE NUMBER
                                ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                ;       RETURN

023666  010246                  SDETAL: MOV     R2,-(SP)                ;SAVE R2
023670  010002                          MOV     R0,R2                   ;DPB ADDRESS
023672  062702  000036                  ADD     #$OPERC,R2              ;FIRST STATISTICAL FIELD
023676  010446                          MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
023700  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
023702     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
023703     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
023704  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
023710  016046  000070                  MOV     $PASSC(R0),-(SP)  ;PUT THE PASS COUNT ON THE STACK
023714  004737  031524                  JSR     PC,$SB2D                ;CONVERT IT
023720  004537  031034                  JSR     R5,REPLZ                ;TYPE IT
023724  000003                          .WORD   3                       ;TYPE 3 DIGITS
023726  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
023732  010246                          MOV     R2,-(SP)                ;PUT $OPERC ON THE STACK
023734  004737  035212                  JSR     PC,$DB2D                ;CONVERT IT
023740  004537  031034                  JSR     R5,REPLZ                ;TYPE $OPERC
023744  000006                          .WORD   6                       ;TYPE 6 DIGITS
023746  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
023752  062702  000004                  ADD     #4,R2                   ;INCREMENT R2
023756  010246                          MOV     R2,-(SP)                ;PUT $POSIT ON THE STACK
023760  004737  035212                  JSR     PC,$DB2D                ;CONVERT IT
023764  004537  031034                  JSR     R5,REPLZ                ;TYPE $POSIT
023770  000006                          .WORD   6                       ;TYPE 6 DIGITS
023772  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
023776  062702  000004                  ADD     #4,R2                   ;INCREMENT R2
024002  010246                          MOV     R2,-(SP)                ;PUT $TRANS ON THE STACK
024004  004737  035212                  JSR     PC,$DB2D                ;CONVERT $TRANS
024010  004537  031034                  JSR     R5,REPLZ                ;TYPE IT
024014  000012                          .WORD   10.                     ;TYPE 10 DIGITS
024016  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
024022  062702  000004                  ADD     #4,R2                   ;INCREMENT R2
024026  010246                          MOV     R2,-(SP)                ;PUT $READ ON THE STACK
024030  004737  035212                  JSR     PC,$DB2D                ;CONVERT $READ
024034  004537  031034                  JSR     R5,REPLZ                ;TYPE IT
024040  000012                          .WORD   10.                     ;TYPE 10 DIGITS
```

```
     024042  104401  073271                TYPE    ,BLNKS1            ;TYPE 1 BLANK
     024046  062702  000004                ADD     #4,R2              ;INCREMENT R2
     024052  062702  000002                ADD     #2,R2              ;INCREMENT R2 AGAIN
     024056  012246                        MOV     (R2)+,-(SP)        ;PUT $SOFT ON THE STACK
     024060  004737  031524                JSR     PC,$SB2D           ;CONVERT $SOFT
     024064  004537  031034                JSR     R5,REPLZ           ;TYPEOUT $SOFT
     024070  000004                        .WORD   4                  ;TYPE 4 DIGITS
     024072  104401  073271                TYPE    ,BLNKS1            ;TYPE 1 BLANK
     024076  012246                        MOV     (R2)+,-(SP)        ;PUT $HARD ON THE STACK
     024100  004737  031524                JSR     PC,$SB2D           ;CONVERT $HARD
     024104  004537  031034                JSR     R5,REPLZ           ;TYPEOUT $HARD
     024110  000004                        .WORD   4                  ;TYPE 4 DIGITS
     024112  104401  073271                TYPE    ,BLNKS1            ;TYPE 1 BLANK
     024116  012246                        MOV     (R2)+,-(SP)        ;PUT $SKI ON THE STACK
     024120  004737  031524                JSR     PC,$SB2D           ;CONVERT $SKI
     024124  004537  031034                JSR     R5,REPLZ           ;TYPEOUT $SKI
     024130  000004                        .WORD   4                  ;TYPE 4 DIGITS
     024132  104401  073271                TYPE    ,BLNKS1            ;TYPE 1 BLANK
     024136  012246                        MOV     (R2)+,-(SP)        ;PUT $MISPO ON THE STACK
     024140  004737  031524                JSR     PC,$SB2D           ;CONVERT $MISPO
     024144  004537  031034                JSR     R5,REPLZ           ;TYPEOUT $MISPO
     024150  000004                        .WORD   4                  ;TYPE 4 DIGITS
     024152  104401  073271                TYPE    ,BLNKS1            ;TYPE 1 BLANK
     024156  016046  000056                MOV     $TOTAL(R0),-(SP)    ;CALCULATE NUMBER OF OTHER ERRORS
     024162  166016  000060                SUB     $SOFT(R0),(SP)     ;SUBTRACT $SOFT FROM $TOTAL
     024166  166016  000062                SUB     $HARD(R0),(SP)     ;SUBTRACT $HARD FROM $TOTAL
     024172  166016  000064                SUB     $SKI(R0),(SP)      ;SUBTRACT $SKI FROM $TOTAL
     024176  166016  000066                SUB     $MISPO(R0),(SP)    ;SUBTRACT $MISPO FROM $TOTAL
     024202  004737  031524                JSR     PC,$SB2D           ;CONVERT 'OTHER' COUNT
     024206  004537  031034                JSR     R5,REPLZ           ;TYPE IT
     024212  000004                        .WORD   4                  ;TYPE 4 DIGITS
     024214  012602                        MOV     (SP)+,R2           ;;POP STACK INTO R2
     024216  000207                        RTS     PC



                                   ;ROUTINE TO INCREMENT $SOFT
                                   ;
                                   ;NOTE:  $SOFT WILL NOT BE INCREMENTED BEYOND 9999 (10)

     024220  005737  001364        INCSOF: TST     BADSEC             ;SEE IF BAD TRK/SEC INDICATOR SET
     024224  001006                        BNE     1$                 ;BR IF IT'S SET, DON'T INCREMENT COUNT
     024226  026027  000060  023417        CMP     $SOFT(R0),#9999.    ;IS $SOFT ALREADY AT MAXIMUM ?
     024234  103002                        BHIS    1$                 ;BR IF IT IS
     024236  005260  000060                INC     $SOFT(R0)                  ;INCREMENT $SOFT
     024242  000207                1$:     RTS     PC                 ;RETURN


                                   ;ROUTINE TO INCREMENT $HARD
                                   ;
                                   ;NOTE:  $HARD WILL NOT BE INCREMENTED BEYOND 9999 (10)

     024244  005737  001364        INCHRD: TST     BADSEC             ;SEE IF BAD TRK/SEC INDICATOR SET
     024250  001006                        BNE     1$                 ;BR IF IT'S SET, DON'T INCREMENT COUNT
     024252  026027  000062  023417        CMP     $HARD(R0),#9999.    ;IS $HARD ALREADY AT MAXIMUM ?
     024260  103002                        BHIS    1$                 ;BR IF IT IS
     024262  005260  000062                INC     $HARD(R0)                  ;INCREMENT $HARD
     024266  000207                          1$:     RTS     PC                      ;RETURN


                                             ;ROUTINE TO INCREMENT $SKI
                                             ;
                                             ;NOTE:  $SKI WILL NOT BE INCREMENTED BEYOND 9999 (10)

     024270  005737  001364                  INCSKI: TST     BADSEC                  ;SEE IF BAD TRK/SEC INDICATOR SET
     024274  001006                                  BNE     1$                      ;BR IF IT'S SET, DON'T INCREMENT COUNT
     024276  026027  000064  023417                  CMP     $SKI(R0),#9999.   ;IS $SKI ALREADY AT MAXIMUM ?
     024304  103002                                  BHIS    1$                      ;BR IF IT IS
     024306  005260  000064                          INC     $SKI(R0)                     ;INCREMENT $SKI
     024312  000207                          1$:     RTS     PC                      ;RETURN


                                             ;ROUTINE TO INCREMENT $MISPO
                                             ;
                                             ;NOTE:  $MISPO WILL NOT BE INCREMENTED BEYOND 9999 (10)

     024314  005737  001364                  INCMIS: TST     BADSEC                  ;SEE IF BAD TRK/SEC INDICATOR SET
     024320  001006                                  BNE     1$                      ;BR IF IT'S SET, DON'T INCREMENT COUNT
     024322  026027  000066  023417                  CMP     $MISPO(R0),#9999.   ;IS $MISPO ALREADY AT MAXIMUM ?
     024330  103002                                  BHIS    1$                      ;BR IF IT IS
     024332  005260  000066                          INC     $MISPO(R0)                   ;INCREMENT $MISPO
     024336  000207                          1$:     RTS     PC                      ;RETURN


                                             ;ROUTINE TO INCREMENT $TOTAL
                                             ;
                                             ;NOTE:  $TOTAL WILL NOT BE INCREMENTED BEYOND 9999 (10)

     024340  005737  001364                  INCTOT: TST     BADSEC                  ;SEE IF BAD TRK/SEC INDICATOR SET
     024344  001006                                  BNE     1$                      ;BR IF IT'S SET, DON'T INCREMENT COUNT
     024346  026027  000056  023417                  CMP     $TOTAL(R0),#9999.   ;IS $TOTAL ALREADY AT MAXIMUM ?
     024354  103002                                  BHIS    1$                      ;BR IF IT IS
     024356  005260  000056                          INC     $TOTAL(R0)                   ;INCREMENT $TOTAL
     024362  000207                          1$:     RTS     PC                      ;RETURN


                                             ;ROUTINE TO TYPE THE TIME

     024364  005737  001312                  $TIME:  TST     CLKFLG                  ;CLOCK ON THE SYSTEM ?
     024370  001033                                  BNE     1$                      ;BR IF NOT
     024372  104401  001203                          TYPE    ,$CRLF                  ;CR-LF
     024376  013746  001366                          MOV     HOUR,-(SP)              ;PUT 'HOURS' ON THE STACK
     024402  004737  031524                          JSR     PC,$SB2D                ;CONVERT TO DECIMAL
     024406  004537  031034                          JSR     R5,REPLZ                ;TYPE IT
     024412  000002                                  .WORD   2                       ;TYPE 2 DIGITS
     024414  104401  074464                          TYPE    ,COLON                  ;':'
     024420  013746  001370                          MOV     MINUTE,-(SP)            ;PUT 'MINUTES' ON THE STACK
     024424  004737  031524                          JSR     PC,$SB2D                ;CONVERT TO DECIMAL
     024430  004537  031034                          JSR     R5,REPLZ                ;TYPE IT
     024434  000002                                  .WORD   2                       ;TYPE 2 DIGITS
     024436  104401  074464                          TYPE    ,COLON                  ;':'
     024442  013746  001372                          MOV     SECOND,-(SP)            ;PUT SECONDS ON THE STACK
     024446  004737  031524                          JSR     PC,$SB2D                ;CONVERT TO DECIMAL
     024452  004537  031034                          JSR     R5,REPLZ                ;TYPE IT
```

```
024456  000002                                  .WORD   2                  ;TYPE 2 DIGITS
024460  000207                          1$:     RTS     PC


                                        ;CLOCK HANDLER ROUTINE


024462  005337  001374                  CLOCK:  DEC     SIXTEE             ;INCREMENT THE 1/60 SECOND COUNTER
024466  001035                                  BNE     1$                 ;BR IF A SECOND NOT COUNTED
024470  013737  001314  001374                  MOV     HZ,SIXTEE          ;RESTORE THE VALUE
024476  005237  001372                          INC     SECOND             ;COUNT THE SECOND
024502  022737  000074  001372                  CMP     #60.,SECOND        ;AT MAXIMUM ?
024510  001024                                  BNE     1$                 ;BR IF NOT
024512  005037  001372                          CLR     SECOND             ;CLEAR THE SECOND'S COUNTER
024516  005237  001474                          INC     INTRVL+2           ;COUNT THE PERFORMANCE SUMMARY INTERVAL
024522  005237  001370                          INC     MINUTE             ;COUNT THE MINUTE
024526  022737  000074  001370                  CMP     #60.,MINUTE        ;AT MAXIMUM ?
024534  001012                                  BNE     1$                 ;BR IF NOT
024536  005037  001370                          CLR     MINUTE             ;CLEAR THE MINUTE'S COUNTER
024542  005237  001366                          INC     HOUR               ;COUNT THE HOURS
024546  022737  001747  001366                  CMP     #999.,HOUR         ;AT MAXIMUM
024554  103002                                  BHIS    1$                 ;BR IF NOT
024556  005037  001366                          CLR     HOUR               ;CLEAR THE HOURS
024562  012746  000020                  1$:     MOV     #20,-(SP)          ;1 MS ON THE STACK
024566  004737  042310                          JSR     PC,RMTMR           ;DRIVER TIMER ROUTINE
024572  005737  001472                          TST     INTRVL             ;DISPLAY THE PERFORMANCE SUMMARY ?
024576  001411                                  BEQ     2$                 ;BR IF NOT
024600  023737  001472  001474                  CMP     INTRVL,INTRVL+2    ;DISPLAY INTERVAL FINISHED ?
024606  001005                                  BNE     2$                 ;BR IF NOT
024610  012737  177777  001316                  MOV     #-1,STATIN         ;SET PERFORMANCE SUMMARY DISPLAY FLAG
024616  005037  001474                          CLR     INTRVL+2           ;CLEAR THE PERFORMANCE INTERVAL COUNTER
024622  000002                          2$:     RTI

                                        ;COMMAND DECODE ROUTINE
                                        ;CALL:
                                        ;       MOV     #-1,CFLAG          ;'CFLAG' IS NORMALLY SET BY THE TTY SERVICE
                                        ;                                  ;ROUTINE IN INTERRUPT MODE
                                        ;       JSR     PC,KSR
                                        ;       RETURN1                    ;SYSTEM BUSY RETURN
                                        ;       RETURN2                    ;RETURN AFTER KEYBOARD SERVICED

024624  005737  001544                  KSR:    TST     ORDERQ+16          ;ANY OPERATIONS ACTIVE ?
024630  001402                                  BEQ     KSR1
024632  104401  074567                          TYPE    ,BUSY              ;'SYSTEM BUSY...'
024636  104412                          KSR1:   SAVREG                     ;SAVE THE REGISTERS
024640  012737  000200  177776                  MOV     #PR4,PS            ;SET PRIORITY TO 4
024646  005037  001362                  1$:     CLR     CFLAG              ;CLEAR THE 'CONTROL C' FLAG
024652  004737  024364                          JSR     PC,$TIME           ;TYPE THE TIME
024656  005777  154300                          TST     @$TKB              ;CLEAR ANY GARBAGE IN THE TTY BUFFER
024662  104401  074332                          TYPE    ,ENTCOM            ;'ENTER COMMAND'

024666  104411                                  RDLIN                      ;READ THE KEYBOARD
024670  012605                                  MOV     (SP)+,R5           ;GET ADDRESS OF INPUT STRING
024672  005737  001362                          TST     CFLAG              ;CHECK THE CONTROL C FLAG
024676  001122                                  BNE     11$                ;EXIT IF 'CONTROL C' ENTERED
024700  005205                                  INC     R5                 ;POINT TO SECOND CHARACTER
024702  122715  000124                          CMPB    #'T,(R5)           ;EQ TO A 'T' ?
024706  001462                                  BEQ     7$                 ;YES
024710  122715  000101                          CMPB    #'A,(R5)           ;EQ TO AN 'A'
```

```
024714  001410                                  BEQ     2$              ;BR IF IT IS
024716  121527  000067                          CMPB    (R5),#'7        ;DRIVE NUMBER GREATER THAN AN ASCII 7 ?
024722  101105                                  BHI     10$             ;BR IF IT IS
024724  121527  000060                          CMPB    (R5),#'0        ;DRIVE NUMBER LESS THAN AN ASCII 0 ?
024730  103502                                  BLO     10$             ;BR IF IT IS
024732  142715  177770                          BICB    #^C7,(R5)       ;LEAVE ONLY LOWER 3 BITS IF CHAR NOT 'A'
024736  122765  000124  177777  2$:             CMPB    #'T,-1(R5)      ;EQ TO 'T'
024744  001003                                  BNE     3$              ;BR IF NOT EQ
024746  004737  025574                          JSR     PC,NEWASN       ;ASSIGN DRIVE FOR TEST
024752  000474                                  BR      11$             ;EXIT
024754  122765  000104  177777  3$:             CMPB    #'D,-1(R5)      ;EQ TO 'D' ?
024762  001003                                  BNE     4$              ;BR IF NOT EQ
024764  004737  025604                          JSR     PC,DEASGN       ;DEASSIGN DRIVE
024770  000465                                  BR      11$             ;EXIT
024772  122765  000123  177777  4$:             CMPB    #'S,-1(R5)      ;EQ TO 'S'
025000  001003                                  BNE     5$              ;BR IF NOT EQ
025002  004737  025704                          JSR     PC,SCMND        ;TYPE STATISTICS
025006  000456                                  BR      11$             ;EXIT
025010  122765  000127  177777  5$:             CMPB    #'W,-1(R5)      ;EQ TO 'W'
025016  001007                                  BNE     6$              ;BR IF NOT EQ
025020  032777  000001  154126                  BIT     #SW0,@SWR       ;IS SWITCH 0 SET ?
025026  001040                                  BNE     9$              ;BR IF SET, CAN'T DO 'W' COMMAND
025030  004737  026132                          JSR     PC,DATAPK       ;WRITE A DATA PACK
025034  000443                                  BR      11$             ;EXIT
025036  122765  000122  177777  6$:             CMPB    #'R,-1(R5)      ;EQ TO 'R' ?
025044  001034                                  BNE     10$             ;BR IF NOT EQ
025046  004737  026162                          JSR     PC,REDAPK       ;READ A DATA PACK
025052  000434                                  BR      11$             ;EXIT
025054  122765  000127  177777  7$:             CMPB    #'W,-1(R5)      ;WT COMMAND ?
025062  001025                                  BNE     10$             ;NO
025064  122765  000101  000001                  CMPB    #'A,1(R5)       ;ALL DRIVES ?
025072  001413                                  BEQ     8$              ;YES
025074  126527  000001  000067                  CMPB    1(R5),#'7       ;GREAT THAN 7
025102  101015                                  BHI     10$             ;YES
025104  126527  000001  000060                  CMPB    1(R5),#'0       ;LESS THAN 0
025112  103411                                  BLO     10$             ;YES
025114  142765  177770  000001                  BICB    #^C7,1(R5)      ;CHOP OFF THE HIGHER BITS
025122  004737  026144          8$:             JSR     PC,WATPAK       ;ASSIGN DRIVES WITH WT COMMAND
025126  000406                                  BR      11$
025130  104401  074252          9$:             TYPE    ,MSWR0          ;TYPE 'CAN'T WRITE WITH SW00 SET'
025134  000644                                  BR      1$              ;TRY AGAIN
025136  104401  074307          10$:            TYPE    ,INVLD          ;TYPE 'INVALID COMMAND' MESSAGE
025142  000641                                  BR      1$              ;TRY AGAIN
025144  104413                  11$:            RESREG                  ;RESTORE R0 - R5
025146  005777  154010                          TST     @$TKB           ;CLEAR THE TTY BUFFER
025152  052777  000100  154000                  BIS     #BIT06,@$TKS    ;SET TTY INTERRUPT ENABLE
025160  005037  177776                          CLR     PS              ;SET PRIORITY BACK TO ZERO
025164  000207                                  RTS     PC              ;RETURN

                                        ;ROUTINE TO PROCESS THE ASSIGN REQUEST ('T', 'R', OR 'W' COMMANDS)

025166  122715  000101          ASSIGN: CMPB    #'A,(R5)        ;ASSIGN ALL DRIVES?
025172  001432                          BEQ     ASGN2           ;BR IF ALL DRIVES
025174  111504                  ASGN1:  MOVB    (R5),R4         ;PUT DRIVE # IN R4
025176  012737  073402  030052          MOV     #UNTASN,ASNMSG  ;'DRIVE ASSIGNED' MESSAGE ADDRESS
025204  005737  001452                  TST     XXDP            ;LOADED FROM THIS DEVICE ?
025210  001412                          BEQ     1$              ;BR IF NO
```

```
025212  123704  001452                          CMPB    XXDP,R4                 ;LOADED FROM THIS DRIVE ?
025216  001007                                  BNE     1$                      ;BR IF NO
025220  146437  035750  001550                  BICB    ATABIT(R4),ASNLST  ;DELETE THE DRIVE FROM THE ASSIGNED LIST
025226  012737  073515  030052                  MOV     #LODEV,ASNMSG   ;'DRIVE IS LOAD DEVICE' MESSAGE ADDRESS
025234  000407                                  BR      2$
025236  136437  035750  001550  1$:             BITB    ATABIT(R4),ASNLST  ;DRIVE ALREADY ASSIGNED ?
025244  001003                                  BNE     2$                      ;BR IF IT IS
025246  004737  025356                          JSR     PC,ASGN3                ;SEE IF DRIVE ON THE SYSTEM
025252  000207                                  RTS     PC                      ;RETURN
025254  000137  030026                  2$:     JMP     ASNERR                  ;EXIT ERROR

025260  005004                          ASGN2:  CLR     R4                      ;START WITH DRIVE 0
025262  012737  073402  030052          1$:     MOV     #UNTASN,ASNMSG  ;ERROR MESSAGE
025270  005737  001452                          TST     XXDP                    ;LOADED FROM THIS DEVICE ?
025274  001412                                  BEQ     2$                      ;BR IF NO
025276  123704  001452                          CMPB    XXDP,R4                 ;LOADED FROM THIS DRIVE ?
025302  001007                                  BNE     2$                      ;BR IF NO
025304  146437  035750  001550                  BICB    ATABIT(R4),ASNLST  ;DELETE THE DRIVE FROM THE ASSIGNED LIST
025312  012737  073515  030052                  MOV     #LODEV,ASNMSG   ;'DRIVE IS LOAD DEVICE' MESSAGE ADDRESS
025320  000413                                  BR      4$
025322  136437  035750  001550  2$:             BITB    ATABIT(R4),ASNLST          ;ALREADY ASSIGNED ?
025330  001007                                  BNE     4$                      ;YES
025332  004737  025356                          JSR     PC,ASGN3                ;ASSIGN THE DRIVE
025336  005204                          3$:     INC     R4                      ;INCREMENT DRIVE #
025340  020427  000007                          CMP     R4,#7                   ;ALL DRIVE CHECKED ?
025344  003746                                  BLE     1$                      ;NO
025346  000207                                  RTS     PC                      ;YES
025350  004737  030026                  4$:     JSR     PC,ASNERR               ;ERROR MESSAGE
025354  000770                                  BR      3$                      ;TO LOOP

025356  136437  035750  001550  ASGN3:  BITB    ATABIT(R4),ASNLST  ;DRIVE ALREADY ASSIGNED ?
025364  001054                                  BNE     ASGN4                   ;BR IF IT IS
025366  110437  066070                          MOVB    R4,GENDPB               ;GET DRIVE NUMBER
025372  006304                                  ASL     R4                      ;MAKE R4 WORD INDEX
025374  016400  002106                          MOV     BLKADR(R4),R0   ;PUT BLOCK'S ADDR INTO R0
025400  004737  015652                          JSR     PC,RECAL0               ;RECALIBRATE DRIVE
025404  006204                                  ASR     R4                      ;MAKE R4 BYTE INDEX
025406  105764  035634                          TSTB    DRVSTA(R4)              ;DRIVE AVAILABLE?
025412  001447                                  BEQ     ASGN7                   ;BR IF DRIVE OFFLINE OR NONEXISTENT
025414  100441                                  BMI     ASGN6                   ;BR IF DRIVE UNSAFE
025416  004737  026174                          JSR     PC,CLRDPB               ;CLEAR BLOCK FOR DRIVE JUST ASSIGNED
025422  104401  074136                          TYPE    ,DRNUM                  ;TYPE DRIVE MESSAGE
025426  010446                                  MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
025430  104403                                  TYPOS                           ;;GO TYPE--OCTAL ASCII
025432     002                                  .BYTE   2                       ;;TYPE 2 DIGIT(S)
025433     000                                  .BYTE   0                       ;;SUPPRESS 'EADING ZEROS
025434  104401  001203                          TYPE    ,$CRLF                  ;CR-LF
025440  006304                                  ASL     R4                      ;MAKE R4 WORD INDEX
025442  004737  026604                          JSR     PC,GETID                ;GET DRIVE I.D.
025446  004737  026374                          JSR     PC,DRVPRM               ;GET THE DRIVE'S ADDRESS LIMITS
025452  004537  026700                          JSR     R5,GETADR               ;RETRIEVE BAD SPOT FILE
025456  000416                                  BR      2$                      ;UNSUCCESSFUL !
025460  004737  027226                          JSR     PC,MANTER               ;MANUALLY ENTER BAD SECTOR INFORMATION
025464  016464  002106  001574                  MOV     BLKADR(R4),NEWUNT(R4) ;DPB ADDRESS
025472  012760  000001  000070                  MOV     #1,$PASSC(R0)   ;PRESET PASS COUNT TO 1
025500  005737  001320                          TST     PACK                    ;WRITE DATA PACK ?
025504  001403                                  BEQ     2$                      ;BR IF NOT
```

```
  2698 025506  113760  001320  000026            MOVB    PACK,$PACK(R0)  ;SET READ/WRITE DATA PACK INDICATOR
  2699 025514  006204                    2$:     ASR     R4              ;MAKE R4 BYTE INDEX
  2700 025516  000207                    ASGN4:  RTS     PC              ;RETURN
  2701
  2702 025520  012737  073505  030052    ASGN6:  MOV     #NOTSAF,ASNMSG  ;'UNSAFE' MESSAGE ADDRESS
  2703 025526  000137  030026                    JMP     ASNERR          ;TO ERROR ROUTINE
  2704
  2705 025532  105764  035644            ASGN7:  TSTB    DRVTYP(R4)      ;DRIVE PRESENT?
  2706 025536  001405                            BEQ     1$              ;BR IF NOT
  2707 025540  100010                            BPL     2$              ;BR IF DRIVE OFFLINE
  2708 025542  012737  073430  030052            MOV     #NOTRM,ASNMSG   ;ADDRESS OF 'NOT RM05/3/2' MSG
  2709 025550  000407                            BR      3$              ;EXIT
  2710 025552  012737  073451  030052    1$:     MOV     #NOTPRS,ASNMSG  ;ADDRESS OF 'NOT PRESENT' MSG
  2711 025560  000403                            BR      3$              ;EXIT
  2712 025562  012737  073337  030052    2$:     MOV     #UNTOFF,ASNMSG  ;ADDRESS OF 'DRIVE OFFLINE' MESSAGE
  2716 025570  000137  030026            3$:     JMP     ASNERR          ;TO ERROR ROUTINE
  2717
  2718                                   ;'T' COMMAND (ROUTINE TO ASSIGN A DRIVE)
  2719
  2720 025574  005037  001320            NEWASN: CLR     PACK            ;CLEAR 'W' COMMAND INDICATOR
  2721 025600  000137  025166                    JMP     ASSIGN          ;GO TO THE ASSIGN ROUTINE
  2722
  2723                                   ;'D' COMMAND (ROUTINE TO DEASSIGN A DRIVE)
  2724
  2725 025604  005004                    DEASGN: CLR     R4              ;START WITH DRIVE 0
  2726 025606  012703  000010                    MOV     #8.,R3          ;COUNTER
  2727 025612  122715  000101                    CMPB    #'A,(R5)        ;DEASSIGN ALL DRIVES ?
  2728 025616  001403                            BEQ     1$              ;BR IF YES
  2729 025620  111504                            MOVB    (R5),R4         ;GET DRIVE NUMBER
  2730 025622  012703  000001                    MOV     #1,R3           ;SET R3 FOR ONE UNIT
  2731 025626  136437  035750  001550    1$:     BITB    ATABIT(R4),ASNLST  ;DRIVE ASSIGNED ?
  2732 025634  001414                            BEQ     3$              ;BR IF NOT
  2733 025636  146437  035750  001550            BICB    ATABIT(R4),ASNLST  ;DELETE THE DRIVE FROM THE ASSIGNED LIST
  2734 025644  006304                            ASL     R4              ;MAKE ADDR INTO A WORD INDEX
  2735 025646  016464  002106  001552            MOV     BLKADR(R4),DUNIT(R4)  ;PUT ADDRESS IN DEASSIGN LIST
  2736 025654  006204                            ASR     R4
  2737 025656  005303                    2$:     DEC     R3              ;ANY MORE DRIVES ?
  2738 025660  001410                            BEQ     4$              ;BR IF NOT
  2739 025662  005204                            INC     R4
  2740 025664  000760                            BR      1$
  2741 025666  012737  073360  030052    3$:     MOV     #UNTNOT,ASNMSG  ;ADDR OF 'NOT ASSIGNED' MESSAGE
  2742 025674  004737  030026                    JSR     PC,ASNERR       ;REPORT IT
  2743 025700  000766                            BR      2$
  2744 025702  000207                    4$:     RTS     PC
  2745
  2746                                   ;'S' COMMAND (ROUTINE TO TYPE DRIVE PERFORMANCE SUMMARY)
  2747
  2748 025704  005004                    SCMND:  CLR     R4
  2749 025706  012703  000010                    MOV     #8.,R3          ;COUNTER
  2750 025712  122715  000101                    CMPB    #'A,(R5)        ;ALL STATISTICS ?
  2751 025716  001421                            BEQ     2$              ;BR IF YES
  2752 025720  111504                            MOVB    (R5),R4         ;GET DRIVE NUMBER
  2753 025722  136437  035750  001550            BITB    ATABIT(R4),ASNLST  ;SEE IF DRIVE ASSIGNED
  2754 025730  001406                            BEQ     1$              ;BR IF NOT
  2755 025732  006304                            ASL     R4              ;MAKE DRIVE ADDR INTO WORD INDEX
  2756 025734  016400  002106                    MOV     BLKADR(R4),R0   ;ADDR OF BLOCK
  2757 025740  004737  023616                    JSR     PC,SUMARY       ;TYPE DRIVE STATISTICS SUMMARY
```

```
 025744  000471                           BR      10$                 ;EXIT

 025746  012737  073360  030052   1$:     MOV     #UNTNOT,ASNMSG      ;ADDR OF 'NOT ASSIGNED' MSG
 025754  004737  030026                   JSR     PC,ASNERR           ;TYPE ERROR MESSAGE
 025760  000463                           BR      10$                 ;EXIT
 025762  105737  001550           2$:     TSTB    ASNLST              ;ANY DRIVE ASSIGNED ?
 025766  001001                           BNE     3$                  ;YES
 025770  000457                           BR      10$                 ;RETURN
 025772  004737  023530           3$:     JSR     PC,STATPR           ;TYPE ALL STATISTICS
 025776  105737  001322                   TSTB    DATE                ;SEE IF 'DATE' ENTERED
 026002  001404                           BEQ     4$                  ;BR IF NOT
 026004  104401  074466                   TYPE    ,DATEIS             ;'DATE: '
 026010  104401  001322                   TYPE    ,DATE               ;THE OPERATOR ENTERED DATE
 026014  105737  001334           4$:     TSTB    OPERID              ;SEE IF OPERATOR I.D. ENTERED
 026020  001404                           BEQ     5$                  ;BR IF NOT
 026022  104401  074476                   TYPE    ,IDIS               ;'OPERATOR I.D.: '
 026026  104401  001334                   TYPE    ,OPERID             ;THE OPERATOR I.D.
 026032  104401  074517           5$:     TYPE    ,HEDLIN             ;HEADER LINE
 026036  012737  046316  026104           MOV     #DRIVE0+$DRVID,8$   ;DRIVE I.D. FIELD ADDRESS
 026044  136437  035750  001550   6$:     BITB    ATABIT(R4),ASNLST   ;SEE IF DRIVE ASSIGNED
 026052  001417                           BEQ     9$                  ;BR IF NOT ASSIGNED
 026054  010446                           MOV     R4,-(SP)            ;;SAVE R4 FOR TYPEOUT
                                                                      ;;TYPE DRIVE NUMBER
 026056  104403                           TYPOS                       ;;GO TYPE--OCTAL ASCII
 026060     002                           .BYTE   2                   ;;TYPE 2 DIGIT(S)
 026061     000                           .BYTE   0                   ;;SUPPRESS LEADING ZEROS
 026062  104401  073266                   TYPE    ,BLNKS4             ;TYPE 4 BLANKS
 026066  105777  000012                   TSTB    @8$                 ;SEE IF DRIVE I.D. ENTERED
 026072  001003                           BNE     7$                  ;BR IF DRIVE I.D. PRESENT
 026074  104401  074540                   TYPE    ,NONE               ;TYPE 'NONE'
 026100  000404                           BR      9$                  ;CONTINUE
 026102  104401                   7$:     TYPE                        ;TYPE THE DRIVE I.D.
 026104  000000                   8$:     .WORD   0                   ;ADDRESS OF DRIVE I.D. FIELD HERE
 026106  104401  001203                   TYPE    ,$CRLF              ;CR-LF
 026112  005303                   9$:     DEC     R3                  ;DECREMENT THE COUNTER
 026114  003405                           BLE     10$                 ;BR IF AT END
 026116  062737  002166  026104           ADD     #$RMEC2+2,8$        ;INCREMENT THE MESSAGE FIELD ADDRESS
 026124  005204                           INC     R4                  ;INCREMENT DRIVE ADDRESS
 026126  000746                           BR      6$                  ;CONTINUE
 026130  000207                   10$:    RTS     PC

                                  ;'W' COMMAND (ROUTINE TO WRITE A DATA PACK)

 026132  012737  177777  001320   DATAPK: MOV     #-1,PACK            ;SET THE 'W' COMMAND INDICATOR
 026140  000137  025166                   JMP     ASSIGN              ;ASSIGN REQUESTED DRIVE

                                  ;'WT' COMMAND (TO WRITE A PACK AND FOLLOWED BY TEST PACK)

 026144  116515  000001           WATPAK: MOVB    1(R5),(R5)          ;ADJUST DRIVE NUMBER ADDRESS
 026150  012737  177776  001320           MOV     #-2,PACK            ;PACK WRITE COMMAND
 026156  000137  025166                   JMP     ASSIGN              ;JUMP TO ASSIGN ROUTINE

                                  ;'R' COMMAND (ROUTINE TO READ A DATA PACK)

 026162  012737  000001  001320   REDAPK: MOV     #1,PACK             ;SET THE 'READ' INDICATOR
 026170  000137  025166                   JMP     ASSIGN              ;ASSIGN THE REQUESTED DRIVE

```

```
2811                                 ;ROUTINE TO CLEAR THE DPB FOR THE ASSIGNED DRIVE
2812                                 ;CALL:
2813                                 ;       MOV     #DPB,R0         ;DPB ADDRESS
2814                                 ;       JSR     PC,CLRDPB
2815                                 ;       RETURN
2816                                 ;
2817                                 ;R0 = DPB ADDRESS BEFORE CALLING THE ROUTINE
2818
2819 026174                          CLRDPB:
     026174  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
     026176  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
     026200  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
     026202  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2820 026204  010004                          MOV     R0,R4           ;GET THE DPB ADDRESS
2821 026206  062704  000002                  ADD     #2,R4           ;ADDRESS OF FIRST LOCN TO BE CLEARED
2822 026212  012703  000005                  MOV     #5,R3           ;NUMBER OF LOCNS TO BE CLEARED
2823 026216  005024                  1$:     CLR     (R4)+           ;CLEAR THE LOCATION
2824 026220  005303                          DEC     R3              ;DECREMENT THE COUNTER
2825 026222  001375                          BNE     1$              ;BR IF NOT FINISHED
2826 026224  062704  000002                  ADD     #2,R4           ;MOVE THE ADDRESS PAST THE 'REG' ADDR
2827 026230  012703  000070                  MOV     #$NEXT-$REG,R3  ;NUMBER OF LOCNS TO BE CLEARED
2828 026234  005024                  2$:     CLR     (R4)+           ;CLEAR
2829 026236  162703  000002                  SUB     #2,R3           ;DECREMENT THE LOCN COUNTER
2830 026242  001374                          BNE     2$              ;BR IF NOT FINISHED
2831 026244  062704  000014                  ADD     #12.,R4         ;MOVE PAST ADDRESS LIMITS
2832 026250  012703  002044                  MOV     #$RMEC2-MINSEC,R3  ;NUMBER OF LOCNS TO BE CLEARED
2833 026254  005024                  3$:     CLR     (R4)+           ;CLEAR A LOCATION
2834 026256  162703  000002                  SUB     #2,R3           ;DECREMENT THE COUNTER
2835 026262  001374                          BNE     3$              ;BR IF NOT DONE
2836 026264  113760  001522  000024          MOVB    BEGCOD,$CODE(R0)  ;INITIAL COMMAND CODE
2837 026272  013701  001522                  MOV     BEGCOD,R1       ;GET THE ACTUAL OP CODE
2838 026276  116160  002126  000002          MOVB    COMTBL(R1),$COMND(R0)  ;OPERATION CODE
2839 026304  113760  001520  000030          MOVB    BEGPAT,$PATTC(R0)  ;PATTERN CODE
2840 026312  106360  000030                  ASLB    $PATTC(R0)      ;CONVERT CODE TO A TABLE INDEX
2841 026316  013760  001524  000020          MOV     BEGSIZ,$WRDL(R0)  ;BEGINNING RECORD SIZE
2842 026324  013760  001524  000004          MOV     BEGSIZ,$WRDM(R0)  ;VALUE FOR DATA TRANSFER
2843 026332  005460  000004                  NEG     $WRDM(R0)       ;MAKE IT INTO 2'S COMPLEMENT
2844 026336  012760  000400  000022          MOV     #256.,$SSEC(R0)  ;INITIAL VALUE OF SECTOR SIZE
2845 026344  132760  000001  000024          BITB    #1,$CODE(R0)    ;HEADER ORDER ?
2846 026352  001403                          BEQ     4$              ;BR IF NOT
2847 026354  062760  000002  000022          ADD     #2,$SSEC(R0)    ;ADD HEADER SIZE TO SECTOR SIZE
2848 026362                          4$:
     026362  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
     026364  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
     026366  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
     026370  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2849 026372  000207                          RTS     PC              ;RETURN
2850
2851                                 ;ROUTINE TO GET ADDRESS LIMITS FROM THE OPERATOR
2852                                 ;CALL
2853                                 ;       MOV     #DPB,R0         ;DPB ADDRESS
2854                                 ;       JSR     PC,DRVPRM       ;CALL ROUTINE
2855                                 ;
2856                                 ;R0 = DPB ADDRESS BEFORE CALLING THE ROUTINE
2857
2858 026374  010346                  DRVPRM: MOV     R3,-(SP)        ;SAVE R3
2859 026376  010446                          MOV     R4,-(SP)        ;SAVE R4
```

```
2860 026400  005737  000042                   TST     @#42              ;RUNNING UNDER MONITOR CONTROL
2861 026404  001002                           BNE     1$                ;BR IF YES
2862 026406  104401  074377                   TYPE    ,ENTLMT           ;'ENTER ADDRESSES'
2863
2864 026412  062760  177777  000122  1$:      ADD     #-1,$FIRST(R0)    ;SEE IF FIRST TIME STARTED
2865 026420  103421                           BCS     3$                ;BR IF NOT
2866 026422  013760  001444  000106           MOV     CYLIMT,MAXCYL(R0)         ;LOAD MAXIMUM CYLINDER
2867 026430  005060  000110                   CLR     MINCYL(R0)        ;CLEAR MINIMUM CYLINDER
2868 026434  004737  026546                   JSR     PC,GETLMT         ;GET ADDRESS LIMITS
2869 026440  013760  001450  000112           MOV     TRKLMT,MAXTRK(R0)         ;LOAD MAXIMUM TRACK
2870 026446  005060  000114                   CLR     MINTRK(R0)        ;CLEAR MINIMUM TRACK
2871 026452  013760  001446  000116           MOV     SECLMT,MAXSEC(R0)         ;LOAD MAXIMUM SECTOR
2872 026460  005060  000120                   CLR     MINSEC(R0)        ;CLEAR MINIMUM SECTOR
2873 026464  016403  075460          3$:      MOV     TABLE(R4),R3      ;PARAMETER TABLE ADDRESS
2874 026470  013763  001450  000016           MOV     TRKLMT,16(R3)     ;LOAD TRACK LIMIT FOR LAST TRACK
2875 026476  013763  001450  000024           MOV     TRKLMT,24(R3)     ;LOAD TRACK LIMIT FOR STARTING TRACK
2876 026504  005737  000042                   TST     @#42              ;UNDER MONITOR CONTROL ?
2877 026510  001002                           BNE     4$                ;BR IF YES
2878 026512  004737  027702                   JSR     PC,PARENT         ;GET THE DRIVE'S PARAMETERS
2879 026516  116060  000120  000010  4$:      MOVB    MINSEC(R0),$SEC(R0)   ;INITIAL SECTOR VALUE
2880 026524  116060  000114  000011           MOVB    MINTRK(R0),$TRK(R0)   ;INITIAL TRACK VALUE
2881 026532  016060  000110  000012           MOV     MINCYL(R0),$CYL(R0)   ;INITIAL CYLINDER VALUE
2882 026540  012604                           MOV     (SP)+,R4          ;RESTORE R4
2883 026542  012603                           MOV     (SP)+,R3          ;RESTORE R3
2884 026544  000207                           RTS     PC                ;RETURN
2885
2886                                 ;ROUTINE TO GET THE ADDRESS LIMITS FOR THE CURRENT DRIVE TYPE
2887                                 ;CALL
2888                                 ;        JSR     PC,GETLMT         ;CALL ROUTINE
2889                                 ;
2890                                 ;R0 = DPB ADDRESS BEFORE CALLING THE ROUTINE
2891
2892 026546                          GETLMT:
     026546  010146                           MOV     R1,-(SP)          ;;PUSH R1 ON STACK
2893 026550  005001                           CLR     R1                ;START FRESH
2894 026552  111001                           MOVB    (R0),R1           ;GET DRIVE NUMBER
2895 026554  012737  000022  001450           MOV     #18.,TRKLMT       ;ASSUME LAST TRACK FOR AN RM05
2896 026562  122761  000007  035644           CMPB    #7,DRVTYP(R1)     ;IS DRIVE AN RM05 ?
2897 026570  001403                           BEQ     1$                ;BR IF YES
2898 026572  012737  000004  001450           MOV     #4,TRKLMT         ;GET LAST TRACK FOR AN RM05
2899 026600                          1$:
     026600  012601                           MOV     (SP)+,R1          ;;POP STACK INTO R1
2900 026602  000207                           RTS     PC                ;RETURN
2901
2902
2903                                 ;ROUTINE TO GET THE DRIVE I.D. FROM THE OPERATOR
2904
2905 026604  010546                  GETID:   MOV     R5,-(SP)          ;SAVE R5
2906 026606  005737  000042                   TST     42                ;UNDER MONITOR CONTROL ?
2907 026612  001030                           BNE     2$                ;BR IF NOT
2908 026614  005037  001362          1$:      CLR     CFLAG             ;CLEAR THE 'CONTROL C' FLAG
2909 026620  104401  074353                   TYPE    ,ENTDRV           ;'ENTER DRV I.D.: '
2910
2911 026624  104411                           RDLIN                     ;READ THE ENTRY
2912 026626  012605                           MOV     (SP)+,R5          ;GET THE ENTRY ADDRESS
2913 026630  005737  001362                   TST     CFLAG             ;'CONTROL C' ENTERED ?
2914 026634  001367                           BNE     1$                ;BR IF IT WAS
```

```
026636  121527  000056                  CMPB    (R5),#'.                ;PERIOD ENTERED ?
026642  001414                          BEQ     2$                      ;BR IF YES
026644  112560  002106                  MOVB    (R5)+,$DRVID(R0)        ;STORE THE DRIVE I.D.
026650  112560  002107                  MOVB    (R5)+,$DRVID+1(R0)
026654  112560  002110                  MOVB    (R5)+,$DRVID+2(R0)
026660  112560  002111                  MOVB    (R5)+,$DRVID+3(R0)
026664  112560  002112                  MOVB    (R5)+,$DRVID+4(R0)
026670  112560  002113                  MOVB    (R5)+,$DRVID+5(R0)
026674  012605                  2$:     MOV     (SP)+,R5                ;RESTORE R5
026676  000207                          RTS     PC                      ;RETURN

                                ;ROUTINE TO GET THE ADDRESSES OF ANY BAD SECTORS IN THE DEC 144
                                ;BAD SECTOR FILE (UP TO A MAX OF 126. FOR MFG PORTION AND
                                ;126. FOR USR PORTION OF DEC 144 FILE.)
                                ;CALL
                                ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                ;       JSR     R5,GETADR
                                ;       RET1                            ERROR RET
                                ;       RET2                            NORMAL RET
                                ;
                                ;R0 = DPB ADDRESS BEFORE CALLING THE ROUTINE

026700                          GETADR:
026700  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
026702  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
026704  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
026706  004737  026546                  JSR     PC,GETLMT               ;GET ADDRESS LIMITS
026712  010001                          MOV     R0,R1                   ;DPB ADDRESS
026714  062701  000124                  ADD     #$BDSEC,R1              ;ADDRESS OF BAD SECTOR TABLE
026720  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
026722  012746  000771                  MOV     #<126.*4>+1,-(SP)       ;;PUSH #<126.*4>+1 ON STACK
026726  011602                          MOV     (SP),R2                 ;NUMBER OF TOTAL ENTRIES AVAILABLE PLUS TERMINATOR
026730  012721  177777          1$:     MOV     #-1,(R1)+               ;INITIALIZE ALL LOCS TO -1
026734  005302                          DEC     R2                      ;DECREMENT WORD
026736  001374                          BNE     1$                      ;BRANCH IF NOT DONE
026740  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
026742  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
026744  111037  066070                  MOVB    (R0),GENDPB             ;DRIVE NUMBER
026750  012737  001466  066102          MOV     #822.,GENDPB+$CYL               ;LAST CYLINDER
026756  113737  001450  066101          MOVB    TRKLMT,GENDPB+$TRK              ;GET LAST TRACK
026764  112737  000000  066100          MOVB    #0,GENDPB+$SEC          ;GET STARTING SECTOR OF MFG FILE
026772  012737  177400  066074          MOV     #-256.,GENDPB+$WRDM             ;ONE SECTOR WORD COUNT
027000  112737  000171  066072          MOVB    #RDDAT,GENDPB+$COMND            ;READ DATA COMMAND
027006  012737  000010  001270          MOV     #8.,$CDW2               ;GET LAST SECTOR OF MFG FILE
027014  012703  076746          2$:     MOV     #CYLNDR,R3              ;GET READ BUFFER ADDRESS
027020  004037  036550                  JSR     R0,RM05                 ;READ CURRENT SECTOR
027024  066070                          GENDPB
027026  000772                          BR      2$                      ;WAIT FOR QUE
027030  005737  066106          3$:     TST     GENDPB+$STATUS          ;READ DONE YET ?
027034  001775                          BEQ     3$                      ;BR IF NO
027036  100010                          BPL     4$                      ;BR IF NO ERROR, ELSE
027040  062737  000002  066100          ADD     #2,GENDPB+$SEC          ;INCREMENT NEXT SECTOR TO READ
027046  123737  001270  066100          CMPB    $CDW2,GENDPB+$SEC               ;WERE ALL SECTORS TRIED ?
027054  103357                          BHIS    2$                      ;BR IF NO
027056  000440                          BR      9$                      ;BR IF UNSUCCESSFUL ON RETRIES
027060  005713                  4$:     TST     (R3)                    ;ARE LSB'S OF SERIAL NUMBER VALID ?
027062  100432                          BMI     7$                      ;BR IF NO
```

```
027064  005763  000002                   TST     2(R3)            ;ARE MSB'S OF SERIAL NUMBER VALID ?
027070  100427                           BMI     7$               ;BR IF NO
027072  062323                           ADD     (R3)+,(R3)+      ;IS SERIAL NUMBER ZERO ?
027074  001425                           BEQ     7$               ;BR IF YES
027076  005723                           TST     (R3)+            ;IS 3RD WORD ALL 0'S ?
027100  001023                           BNE     7$               ;BR IF NO
027102  005723                           TST     (R3)+            ;IS 4TH WORD ALL 0'S ? (ALIGNMENT PACK ?)
027104  001022                           BNE     8$               ;BR IF NO
027106  022713  177777           5$:     CMP     #-1,(R3)         ;IS NEXT WORD A TERMINATOR ?
027112  001403                           BEQ     6$               ;BR IF YES
027114  012321                           MOV     (R3)+,(R1)+      ;STORE BAD CYLINDER ADDRESS
027116  012321                           MOV     (R3)+,(R1)+      ;STORE BAD TRK/SEC ADDRESS
027120  000772                           BR      5$
027122  123727  066100  000012   6$:     CMPB    GENDPB+$SEC,#10.         ;USR BAD FILE DONE YET ?
027130  103031                           BHIS    12$              ;BR IF YES
027132  012737  000036  001270           MOV     #30.,$CDW2       ;GET LAST SECTOR OF USR BAD SECTOR FILE
027140  112737  000012  066100           MOVB    #10.,GENDPB+$SEC         ;GET STARTING SECTOR OF USR BAD SECTOR FILE
027146  000722                           BR      2$
027150  005725                   7$:     TST     (R5)+            ;OK TO USE PACK ANYWAY
027152  104401  074774           8$:     TYPE    ,MERR2           ;INVALID FILE STRUCTURE
027156  000402                           BR      10$
027160  104401  074716           9$:     TYPE    ,MERR1           ;FAILS TO RETRIEVE BAD SPOT FILE
027164  104401  073331           10$:    TYPE    ,UNTMSG          ;ON DRIVE
027170  011046                           MOV     (R0),-(SP)       ;;SAVE (R0) FOR TYPEOUT
027172  104403                           TYPOS                    ;;GO TYPE--OCTAL ASCII
027174     002                           .BYTE   2                ;;TYPE 2 DIGIT(S)
027175     000                           .BYTE   0                ;;SUPPRESS LEADING ZEROS
027176  104401  001203                   TYPE    ,$CRLF           ;CR-LF
027202  012721  177777           11$:    MOV     #-1,(R1)+        ;INITIALIZE ALL LOCS TO -1
027206  005302                           DEC     R2               ;DECREMENT WORD CTR
027210  001374                           BNE     11$              ;BRANCH IF NOT DONE
027212  000401                           BR      13$              ;ERROR RETURN
027214  005725                   12$:    TST     (R5)+            ;ADJUST FOR NORMALRETURN
027216                           13$:
027216  012603                           MOV     (SP)+,R3         ;;POP STACK INTO R3
027220  012602                           MOV     (SP)+,R2         ;;POP STACK INTO R2
027222  012601                           MOV     (SP)+,R1         ;;POP STACK INTO R1
027224  000205                           RTS     R5               ;EXIT

                                 ;ROUTINE TO ENTER BAD SECTOR INFORMATION MANUALLY
                                 ;CALL
                                 ;       MOV     #DPB,R0          ;DPB ADDRESS
                                 ;       JSR     PC,MANTER        ;CALL ROUTINE
                                 ;
                                 ;R0 = DPB ADDRESS BEFORE CALLING THE ROUTINE

027226                           MANTER:
027226  010146                           MOV     R1,-(SP)         ;;PUSH R1 ON STACK
027230  010246                           MOV     R2,-(SP)         ;;PUSH R2 ON STACK
027232  010346                           MOV     R3,-(SP)         ;;PUSH R3 ON STACK
027234  010446                           MOV     R4,-(SP)         ;;PUSH R4 ON STACK
027236  105737  001150                   TSTB    $AUTOB           ;RUNNING UNDER AUTO MODE ?
027242  001402                           BEQ     1$               ;BR IF NO
027244  000137  027670                   JMP     21$              ;YES, EXIT
027250  005037  001362           1$:     CLR     CFLAG            ;CLEAR THE CONTROL-C FLAG
027254  104401  074427                   TYPE    ,ENTADR          ;MESSAGE TO ENTER...
```

```
  3012 027260  012704  000124                  MOV     #$BDSEC,R4          ;INDEX VALUE OF TABLE ADDRESS
  3013 027264  060004                          ADD     R0,R4               ;TABLE STARTING ADDRESS
  3014 027266  012701  000374                  MOV     #<126.*2>,R1        ;TOTAL BAD SPOTS ALLOWED
  3015 027272  022714  177777          2$:     CMP     #-1,(R4)            ;ENTRY IN THE TABLE ?
  3016 027276  001407                          BEQ     3$                  ;BRANCH IF SO
  3017 027300  062704  000004                  ADD     #4,R4               ;ADJUST THE TABLE ENTRY POINTER
  3018 027304  005301                          DEC     R1                  ;DECREMENT THE BAD SECTOR COUNT
  3019 027306  001371                          BNE     2$                  ;BR IF TO NEXT ENTRIES POSITION
  3020 027310  104401  075040                  TYPE    ,MSFULL             ;TYPE 'BAD SPOT TABLE IS FULL'
  3021 027314  000565                          BR      21$                 ;EXIT..
  3022
  3023 027316  010146                  3$:     MOV     R1,-(SP)            ;THE COUNTER AND FIRST
  3024 027320  010446                          MOV     R4,-(SP)            ;ENTRY POINTER PAIR
  3025 027322  012714  177777          4$:     MOV     #-1,(R4)            ;RESET CYLINDER TO -1
  3026 027326  104401  075072                  TYPE    ,MSGCYL             ;TYPE 'CYLIND'
  3027 027332  012746  177777                  MOV     #-1,-(SP)           ;;SAVE #-1 FOR TYPEOUT
       027336  104405                          TYPDS                       ;;GO TYPE--DECIMAL ASCII WITH SIGN
  3028 027340  104401  075216                  TYPE    ,SLASH              ;TYPE ' / '
  3029 027344  104411                          RDLIN                       ;READ IN THE BAD SPOTS
  3030 027346  012601                          MOV     (SP)+,R1            ;READ IN TEXT ADDRESS
  3031 027350  005737  001362                  TST     CFLAG               ;CONTROL-C ENTERED ?
  3032 027354  001411                          BEQ     7$                  ;BRANCH IF NOT
  3033 027356  012604                  5$:     MOV     (SP)+,R4            ;RETRIEVE THE FIRST ENTRY POINTER
  3034 027360  012601                          MOV     (SP)+,R1            ;RETRIEVE THE SPOT COUNT
  3035 027362  012724  177777          6$:     MOV     #-1,(R4)+           ;RESET THE TABLE
  3036 027366  012724  177777                  MOV     #-1,(R4)+           ;TO -1
  3037 027372  005301                          DEC     R1                  ;DONE WITH TABLE YET ?
  3038 027374  001372                          BNE     6$                  ;BR IF NO
  3039 027376  000724                          BR      1$                  ;START AGAIN
  3040
  3041 027400                          7$:
       027400  013702  001444                  MOV     CYLIMT,R2           ;UPPER LIMIT OF INPUT
       027404  004537  031366                  JSR     R5,CK.DIG           ;CHECK THE DIGIT(S)
       027410  027664                          20$                         ;CARRIAGE RETURN ONLY ENTERED
       027412  027664                          20$                         ;PERIOD ONLY ENTERED
       027414  027424                          8$                          ;ILLEGAL INPUT
       027416  027444                          10$                         ;TERMINATED WITH A CARRIAGE RETURN
       027420  027424                          8$                          ;TERMINATED WITH A ","
       027422  027432                          9$                          ;TERMINATED WITH A ":"
  3042 027424  104401  074546          8$:     TYPE    ,BADENT             ;TYPE BAD ENTRY MESSAGE
  3043 027430  000734                          BR      4$                  ;AND TRY AGAIN.
  3044 027432  010214                  9$:     MOV     R2,(R4)             ;ENTER CYLINDER ADDRESS
  3045 027434  012764  177777  000002          MOV     #-1,2(R4)           ;RESET TRACK AND SECTOR FIELD
  3046 027442  000510                          BR      20$
  3047 027444  010214                  10$:    MOV     R2,(R4)             ;ENTER CYLINDER ADDRESS
  3048
  3049 027446  012764  177777  000002  11$:    MOV     #-1,2(R4)           ;RESET TRACK AND SECTOR FIELDS TO -1
  3050 027454  104401  075101                  TYPE    ,MSGTRK             ;TYPE 'TRACK'
  3051 027460  012746  177777                  MOV     #-1,-(SP)           ;;SAVE #-1 FOR TYPEOUT
       027464  104405                          TYPDS                       ;;GO TYPE--DECIMAL ASCII WITH SIGN
  3052 027466  104401  075216                  TYPE    ,SLASH              ;TYPE ' / '
  3053 027472  104411                          RDLIN                       ;READ IN THE BAD SPOTS
  3054 027474  012601                          MOV     (SP)+,R1            ;READ IN TEXT ADDRESS
  3055 027476  005737  001362                  TST     CFLAG               ;CONTROL-C ENTERED ?
  3056 027502  001325                          BNE     5$                  ;BR IF YES
  3057 027504  013702  001450                  MOV     TRKLMT,R2           ;UPPER LIMIT OF INPUT
       027510  004537  031366                  JSR     R5,CK.DIG           ;CHECK THE DIGIT(S)
```

```
     027514  027550                            15$                        ;CARRIAGE RETURN ONLY ENTERED
     027516  027664                            20$                        ;PERIOD ONLY ENTERED
     027520  027530                            12$                        ;ILLEGAL INPUT
     027522  027544                            14$                        ;TERMINATED WITH A CARRIAGE RETURN
     027524  027530                            12$                        ;TERMINATED WITH A ","
     027526  027536                            13$                        ;TERMINATED WITH A "."
     027530  104401  074546            12$:    TYPE    ,BADENT            ;TYPE BAD ENTRY MESSAGE
     027534  000744                            BR      11$                ;AND TRY AGAIN.
     027536  110264  000003            13$:    MOVB    R2,3(R4)           ;ENTER TRACK ADDRESS AND
     027542  000450                            BR      20$                ;EXIT
     027544  110264  000003            14$:    MOVB    R2,3(R4)           ;ENTER TRACK ADDRESS

     027550  112764  177777  000002    15$:    MOVB    #-1,2(R4)          ;RESET SECTOR TO -1
     027556  104401  075110                    TYPE    ,MSGSEC            ;TYPE 'SECTOR'
     027562  012746  177777                    MOV     #-1,-(SP)          ;;SAVE #-1 FOR TYPEOUT
     027566  104405                            TYPDS                      ;;GO TYPE--DECIMAL ASCII WITH SIGN
     027570  104401  075216                    TYPE    ,SLASH             ;TYPE ' / '
     027574  104411                            RDLIN                      ;READ IN THE BAD SPOTS
     027576  012601                            MOV     (SP)+,R1           ;READ IN TEXT ADDRESS
     027600  005737  001362                    TST     CFLAG              ;CONTROL-C ENTERED ?
     027604  001264                            BNE     5$                 ;BR IF YES
     027606  013702  001446                    MOV     SECLMT,R2          ;UPPER LIMIT OF INPUT
     027612  004537  031366                    JSR     R5,CK.DIG          ;CHECK THE DIGIT(S)
     027616  027652                            19$                        ;CARRIAGE RETURN ONLY ENTERED
     027620  027664                            20$                        ;PERIOD ONLY ENTERED
     027622  027632                            16$                        ;ILLEGAL INPUT
     027624  027646                            18$                        ;TERMINATED WITH A CARRIAGE RETURN
     027626  027632                            16$                        ;TERMINATED WITH A ","
     027630  027640                            17$                        ;TERMINATED WITH A "."
     027632  104401  074546            16$:    TYPE    ,BADENT            ;TYPE BAD ENTRY MESSAGE
     027636  000744                            BR      15$                ;AND TRY AGAIN.
     027640  110264  000002            17$:    MOVB    R2,2(R4)           ;ENTER SECTOR ADDRESS AND
     027644  000407                            BR      20$                ;EXIT
     027646  110264  000002            18$:    MOVB    R2,2(R4)           ;ENTER SECTOR ADDRESS

     027652  005303                    19$:    DEC     R3                 ;MORE ENTRY ?
     027654  001403                            BEQ     20$                ;BRANCH IF EXHAUSTED
     027656  062704  000004                    ADD     #4,R4              ;ADJUST FOR THE NEXT TABLE ENTRY
     027662  000617                            BR      4$                 ;ENTER NEXT SPOT ADDRESS
     027664  062706  000004            20$:    ADD     #4,SP              ;RESTORE STACK
     027670                            21$:
     027670  012604                            MOV     (SP)+,R4           ;;POP STACK INTO R4
     027672  012603                            MOV     (SP)+,R3           ;;POP STACK INTO R3
     027674  012602                            MOV     (SP)+,R2           ;;POP STACK INTO R2
     027676  012601                            MOV     (SP)+,R1           ;;POP STACK INTO R1
     027700  000207                            RTS     PC                 ;EXIT


                                       ;PARAMETER ENTRY ROUTINE
                                       ;CALL
                                       ;       MOV     #ADR,R3            ;PARAMETER TABLE ADDRESS
                                       ;       JSR     PC,PARENT          ;GET THE PARAMETERS

     027702  010346                    PARENT: MOV     R3,-(SP)           ;SAVE THE PARAMETER TABLE ADDRESS
     027704  005037  001362                    CLR     CFLAG              ;CLEAR THE 'CONTROL C' FLAG
     027710  012337  027720            1$:     MOV     (R3)+,3$           ;ADDRESS OF PARAMETER NAME
     027714  001442                            BEQ     9$                 ;BR IF AT END OF TABLE
     027716  104401                            TYPE                       ;TYPE THE PARAMETER NAME
```

```
027720  000000                   3$:      .WORD   0                   ;ADDRESS OF PARAMETER NAME TEXT
027722  012302                            MOV     (R3)+,R2            ;MAXIMUM PARAMETER VALUE
027724  012305                            MOV     (R3)+,R5            ;ADDRESS OF PARAMETER
027726  011546                            MOV     (R5),-(SP)          ;CURRENT VALUE OF PARAMETER
027730  104405                            TYPDS                       ;TYPE THE CURRENT VALUE OF THE PARAMETER
027732  104401  075216                    TYPE    ,SLASH              ;' / '
027736  104411                            RDLIN                       ;READ THE KEYBOARD
027740  012601                            MOV     (SP)+,R1            ;INPUT ASCII STRING ADDRESS
027742  005737  001362                    TST     CFLAG               ;'CONTROL C' ENTERED ?
027746  001021                            BNE     8$                  ;BR IF IT WAS
027750  004537  031366                    JSR     R5,CK.DIG           ;CHECK THE DIGIT(S)
027754  027710                            1$                          ;CARRIAGE RETURN ONLY ENTERED
027756  030022                            9$                          ;PERIOD ONLY ENTERED
027760  027774                            6$                          ;ILLEGAL INPUT
027762  027770                            5$                          ;TERMINATED WITH A CARRIAGE RETURN
027764  027774                            6$                          ;TERMINATED WITH A '',''
027766  030006                            7$                          ;TERMINATED WITH A ''.''
027770  010215                   5$:      MOV     R2,(R5)             ;MOVE NEW VALUE TO PARAMETER LOCATION
027772  000746                            BR      1$                  ;GET MORE PARAMETERS
027774  104401  074546           6$:      TYPE    ,BADENT             ;'BAD ENTRY'
030000  162703  000006                    SUB     #6,R3               ;DECREMENT THE TABLE POINTER
030004  000741                            BR      1$                  ;TRY AGAIN
030006  010215                   7$:      MOV     R2,(R5)             ;NEW VALUE
030010  000404                            BR      9$                  ;EXIT
030012  005037  001362           8$:      CLR     CFLAG               ;CLEAR THE 'CONTROL C' FLAG
030016  011603                            MOV     (SP),R3             ;RELOAD THE PARAMETER TABLE ADDRESS
030020  000733                            BR      1$                  ;TRY AGAIN
030022  005726                   9$:      TST     (SP)+               ;CORRECT THE STACK POINTER
030024  000207                            RTS     PC                  ;RETURN

                                 ;TYPEOUT ASSIGN/DEASSIGN ERROR MESSAGE
                                 ;CALL
                                 ;        MOV     #MESADR,ASNMSG  ;ERROR MESSAGE ADDRESS
                                 ;        JSR     PC,ASNERR
                                 ;        RETURN

030026  104401  001203           ASNERR:  TYPE    ,$CRLF              ;CR-LF
030032  104401  073327                    TYPE    ,QUES               ;'?'
030036  104401  073331                    TYPE    ,UNTMSG             ;TYPE 'DRIVE'
030042  010446                            MOV     R4,-(SP)            ;;SAVE R4 FOR TYPEOUT
                                                                      ;;TYPE DRIVE NUMBER
030044  104403                            TYPOS                       ;;GO TYPE--OCTAL ASCII
030046     002                            .BYTE   2                   ;;TYPE 2 DIGIT(S)
030047     000                            .BYTE   0                   ;;SUPPRESS LEADING ZEROS
030050  104401                            TYPE                        ;TYPE SPECIFIC MESSAGE
030052  000000                   ASNMSG:  .WORD   0                   ;MESSAGE ADDRESS
030054  000207                            RTS     PC

                                 ;DEASSIGN DRIVE IF A FATAL ERROR OCCURS
                                 ;CALL
                                 ;        JSR     PC,DROP
                                 ;        RETURN

030056  005004                   DROP:    CLR     R4                  ;CLEAR R4 FOR DRIVE NUMBER
030060  111004                            MOVB    (R0),R4             ;MOVE DRIVE NUMBER TO R4
030062  146437  035750  001550            BICB    ATABIT(R4),ASNLST   ;REMOVE DRIVE FROM ASSIGNED LIST
030070  006304                            ASL     R4                  ;MAKE DRIVE NUMBER INTO A TABLE INDEX
```

```
 3144 030072  010064  001552                   MOV     R0,DUNIT(R4)      ;PUT DRIVE IN DROP LIST
 3145 030076  104401  001203                   TYPE    ,$CRLF
 3146 030102  104401  074011                   TYPE    ,DROPNG           ;TYPE 'FATAL OR EXCESSIVE ERRORS'
 3147 030106  104401  074064                   TYPE    ,MSGON            ;TYPE 'ON'
 3148 030112  104401  073331                   TYPE    ,UNTMSG           ;TYPE 'DRIVE'
 3149 030116  006204                           ASR     R4                ;DRIVE NUMBER
 3150 030120  010446                           MOV     R4,-(SP)          ;;SAVE R4 FOR TYPEOUT
                                                                         ;;TYPE DRIVE NUMBER
      030122  104403                           TYPOS                     ;;GO TYPE--OCTAL ASCII
      030124     002                           .BYTE   2                 ;;TYPE 2 DIGIT(S)
      030125     000                           .BYTE   0                 ;;SUPPRESS LEADING ZEROS
 3151 030126  105737  001550                   TSTB    ASNLST            ;MORE DRIVES ACTIVE
 3152 030132  001006                           BNE     1$                ;YES
 3153 030134  005737  000042                   TST     @#42              ;ANY MONITOR ?
 3154 030140  001403                           BEQ     1$                ;NO
 3155 030142  005726                           TST     (SP)+             ;CLEAR STACK
 3156 030144  000137  030526                   JMP     $GET42            ;GIVE CONTROL TO MONITOR
 3157 030150  000207                   1$:     RTS     PC
 3158
 3159                                  ;ROUTINE TO DEASSIGN DRIVE IF ERRORS BECOMES EXCESSIVE
 3160
 3161 030152  032777  000020  150774   ABNRML: BIT     #SW04,@SWR        ;SEE IF SWITCH 4 SET
 3162 030160  001006                           BNE     1$                ;BR IF IT'S SET
 3163 030162  023760  001470  000056           CMP     MAXER,$TOTAL(R0)  ;CHECK TOTAL ERROR VALUE
 3164 030170  103002                           BHIS    1$                ;BR IF ERRORS DO NOT EXCEED MAX
 3165 030172  000137  030056                   JMP     DROP              ;DEASSING THE DRIVE
 3166 030176  000207                   1$:     RTS     PC                ;RETURN
 3167
 3168                                  ;ROUTINE TO CHECK FOR END OF PASS AND END OF TEST
 3169
 3240
 3241                                  .SBTTL  END OF PASS ROUTINE

                                       ;;*******************************************************************
                                       ;*INCREMENT THE PASS NUMBER ($PASS)
                                       ;*IF THERES A MONITOR GO TO IT
                                       ;*IF THERE ISN'T JUMP TO FISK

      030200                           $EOP:
      030200  005737  001512                   TST     ENDET             ;END OF PASS DETERMINED BY SEEKS OR WORDS ?
      030204  001412                           BEQ     EOP1              ;BR IF SEEKS
      030206  026037  000054  001456           CMP     $READ+2(R0),ENDCON+2  ;CHECK MSW OF WORDS READ COUNT
      030214  101017                           BHI     EOP2              ;BR IF MSW GREATER THAN LIMIT
      030216  103527                           BLO     EOPX              ;BR IF MSW LESS THAN LIMIT
      030220  026037  000052  001454           CMP     $READ(R0),ENDCON  ;CHECK LSW AGAINST LIMIT
      030226  103012                           BHIS    EOP2              ;BR IF EQUAL OR GREATER
      030230  000522                           BR      EOPX              ;EXIT

      030232  026037  000044  001462   EOP1:   CMP     $POSIT+2(R0),ENDSEK+2  ;CHECK MSW OF SEEK COUNT
      030240  101005                           BHI     EOP2              ;BR IF MSW GREATER THAN LIMIT
      030242  103515                           BLO     EOPX              ;EXIT IF MSW LESS THAN LIMIT
      030244  026037  000042  001460           CMP     $POSIT(R0),ENDSEK  ;CHECK LSW OF SEEK COUNT
      030252  103511                           BLO     EOPX              ;EXIT IF LSW LESS THAN LIMIT

      030254  104401  001203           EOP2:   TYPE    ,$CRLF            ;CR-LF
      030260  104401  074045                   TYPE    ,ENDPAS           ;END OF PASS FOR THE DRIVE
      030264  016046  000070                   MOV     $PASSC(R0),-(SP)          ;;SAVE $PASSC(R0) FOR TYPEOUT
```

```
030270  104405                          TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
030272  111037  001346                  MOVB    (R0),UNIT               ;STORE THE DRIVE NUMBER
030276  104401  074064                  TYPE    ,MSGON                  ;TYPE 'ON'
030302  104401  073331                  TYPE    ,UNTMSG                 ;'DRIVE '
030306  013746  001346                  MOV     UNIT,-(SP)              ;;SAVE UNIT FOR TYPEOUT
030312  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
030314     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
030315     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
030316  104401  001203                  TYPE    ,$CRLF                  ;CR-LF

030322  032777  000020  150624          BIT     #SW04,@SWR              ;TYPE END OF TEST MESSAGE ?
030330  001017                          BNE     1$                      ;BR IF NO
030332  026037  000070  001464          CMP     $PASSC(R0),PASCNT  ;SEE IF AT END OF TEST
030340  103413                          BLO     1$                      ;BR IF NOT
030342  104401  074071                  TYPE    ,ENDTST                 ;TYPE 'END OF TEST'
030346  104401  074107                  TYPE    ,MSGFOR                 ;TYPE 'FOR'
030352  104401  073331                  TYPE    ,UNTMSG                 ;'DRIVE '
030356  013746  001346                  MOV     UNIT,-(SP)              ;;SAVE UNIT FOR TYPEOUT
030362  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
030364     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
030365     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
030366  000421                          BR      3$                      ;DEASSIGN THE DRIVE

030370  004737  023616          1$:     JSR     PC,SUMARY               ;TYPE THE DRIVE'S STATISTICS SUMMARY
030374  010346                          MOV     R3,-(SP)                ;SAVE R3
030376  010446                          MOV     R4,-(SP)                ;SAVE R4
030400  010004                          MOV     R0,R4                   ;DRIVE'S BLOCK ADDRESS
030402  062704  000036                  ADD     #$OPERC,R4              ;ADD THE STARTING ADDR OF SECTIONS TO CLEAR
030406  012703  000010                  MOV     #8.,R3                  ;NUMBER OF LOCNS TO BE CLEARED
                                                                        ;(ERROR COUNTERS NOT CLEARED)
030412  005024                  2$:     CLR     (R4)+                   ;CLEAR THE LOCN
030414  005303                          DEC     R3                      ;DECREMENT THE LOCATION COUNTER
030416  001375                          BNE     2$                      ;BR IF MORE TO GO
030420  012604                          MOV     (SP)+,R4                ;RESTORE R4
030422  012603                          MOV     (SP)+,R3                ;RESTORE R3
030424  005260  000070                  INC     $PASSC(R0)              ;INCREMENT THE PASS COUNT
030430  000422                          BR      EOPX                    ;EXIT

030432  005004                  3$:     CLR     R4                      ;CLEAR R4 FOR DRIVE NUMBER
030434  111004                          MOVB    (R0),R4                 ;MOVE DRIVE NUMBER
030436  146437  035750  001550          BICB    ATABIT(R4),ASNLST  ;DELETE DRIVE FROM ASSIGNED LIST
030444  006304                          ASL     R4                      ;MAKE DRIVE NUMBER INTO TABLE INDEX
030446  010064  001552                  MOV     R0,DUNIT(R4)            ;PUT BLOCK ADDRESS INTO DROP LIST
030452  105737  001550                  TSTB    ASNLST                  ;ALL DRIVES ARE DESIGNED ?
030456  001007                          BNE     EOPX                    ;BRANCH IF NOT
030460  005237  001216                  INC     $DEVCT                  ;INCREMENT DEVICE COUNT
030464  005237  001214                  INC     $PASS                   ;INCREMENT THE PASS NUMBER FOR APT
030470  042737  100000  001214          BIC     #100000,$PASS           ;AVOID NEGATIVE NUMBER
030476  000207                  EOPX:   RTS     PC                      ;RETURN
030500  005237  001214                  INC     $PASS                   ;;INCREMENT THE PASS NUMBER
030504  042737  100000  001214          BIC     #100000,$PASS           ;;DON'T ALLOW A NEG. NUMBER
030512  005327                          DEC     (PC)+                   ;;LOOP?
030514  000001                  $EOPCT: .WORD   1
030516  003013                          BGT     $DOAGN                  ;;YES
030520  012737                          MOV     (PC)+,@(PC)+            ;;RESTORE COUNTER
030522  000001                  $ENDCT: .WORD   1
030524  030514                          $EOPCT
```

```
030526  013700  000042          $GET42: MOV     @#42,R0                 ;;GET MONITOR ADDRESS
030532  001405                          BEQ     $DOAGN                  ;;BRANCH IF NO MONITOR
030534  000005                          RESET                           ;;CLEAR THE WORLD
030536  004710                  $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
030540  000240                          NOP                             ;;SAVE ROOM
030542  000240                          NOP                             ;;FOR
030544  000240                          NOP                             ;;ACT11
030546                          $DOAGN:
030546  000137                          JMP     @(PC)+                  ;;RETURN
030550  030552                  $RTNAD: .WORD   FISK
030552  005237  001212          FISK:   INC     $TESTN                  ;INCREMENT THE TEST NUMBER IN THE MAIL BOX
030556  000137  006260                  JMP     MAIN1                   ;RETURN TO LOOP

                                ;ROUTINE TO GET THE REMAINDER OF THE RANDOM NUMBER
                                ;CALL
                                ;       MOV     NUMBER,R5               ;DIVISOR INTO R5
                                ;       JSR     PC,GETREM
                                ;       RETURN                          ;REMAINDER IS IN R5

030562  013746  035114          GETREM: MOV     $LONUM,-(SP)            ;STORE RANDOM NUMBER ON THE STACK FOR DIVIDE
030566  013746  035112                  MOV     $HINUM,-(SP)            ;UPPER PART
030572  010546                          MOV     R5,-(SP)                ;PUT THE DIVISOR ONTO THE STACK
030574  004737  030610                  JSR     PC,LINKDV               ;DIVIDE THE RANDOM NUMBERS
030600  012605                          MOV     (SP)+,R5                ;PUT THE REMAINDER INTO R5
030602  005726                          TST     (SP)+                   ;ADJUST THE STACK POINTER
030604  000240                          NOP                             ;FOR DEBUGGING HALT
030606  000207                          RTS     PC

                                ;LINK ROUTINE TO THE DIVISION UTILITY SUBROUTINE
                                ;       THIS ROUTINE ALLOWS THE 'SYSMAC' DIVIDE ROUTINE
                                ;       CALLING SEQUENCE TO BE USED

030610  104412                  LINKDV: SAVREG                          ;STORE R0 - R5
030612  016605  000026                  MOV     26(SP),R5               ;DIVISOR
030616  005004                          CLR     R4                      ;OTHER DIVISOR WORD
030620  016602  000030                  MOV     30(SP),R2               ;UPPER DIVIDEND WORD
030624  016603  000032                  MOV     32(SP),R3               ;LOWER DIVIDEND WORD
030630  005000                          CLR     R0                      ;CLEAR OTHER DIVIDEND REGISTERS
030632  005001                          CLR     R1
030634  004737  030656                  JSR     PC,M.DPID               ;GO TO THE DIVIDE ROUTINE
030640  010166  000030                  MOV     R1,30(SP)               ;REMAINDER ON THE STACK
030644  010366  000032                  MOV     R3,32(SP)               ;QUOTIENT ON THE STACK
030650  104413                          RESREG                          ;RESTORE R0 - R5
030652  012616                          MOV     (SP)+,(SP)              ;MOVE RETURN UP THE STACK
030654  000207                          RTS     PC

                                ;       DIVISION UTILITY SUBROUTINE
                                ;       R0-R1-R2-R3=DIVIDEND
                                ;       R4-R5=DIVISOR
                                ;       R0-R1=REMAINDER AFTER DIVISION
                                ;       R2-R3=QUOTIENT AFTER DIVISION
                                ;       ENTER WITH JSR  PC,M.DPID
                                ;

030656  012746  000040          M.DPID: MOV     #40,-(SP)               ;COUNTER FOR DIVISION CYCLES
030662  010446                          MOV     R4,-(SP)                ;HIGH ORDER
030664  010546                          MOV     R5,-(SP)                ;LOW ORDER DIVISOR TO THE STACK
```

```
030666  005466  000002              NEG     2(SP)                   ;FORM NEGATIVE
030672  005416                      NEG     @SP                     ;VERSION OF THE DIVISOR
030674  005666  000002              SBC     2(SP)
030700  061601                      ADD     @SP,R1
030702  005500                      ADC     R0                      ;PERFORM THE INITIAL SUBTRACTION
030704  066600  000002              ADD     2(SP),R0
030710  103445                      BCS     M.DP50                  ;IF CARRY THEN OVERFLOW HAS OCCURRED
030712  005046                      CLR     -(SP)                   ;THIS IS A LONGER LASTING CARRY BIT
030714  006103            M.DP40:   ROL     R3
030716  006102                      ROL     R2
030720  006101                      ROL     R1
030722  006100                      ROL     R0
030724  005716                      TST     @SP                     ;TEST ''CARRY'' INDICATOR
030726  001410                      BEQ     M.DP41                  ;IF NO ''CARRY'' THEN ADD ELSE SUBTRACT
030730  005016                      CLR     @SP                     ;CLEAR UP FOR NEXT TIME
030732  066601  000002              ADD     2(SP),R1
030736  005500                      ADC     R0                      ;ADD -(DIVISOR)
030740  005516                      ADC     @SP         ;  I        ;SET ''CARRY''
030742  066600  000004              ADD     4(SP),R0;<-
030746  000404                      BR      M.DP42
030750  060501            M.DP41:   ADD     R5,R1
030752  005500                      ADC     R0                      ;ADD +(DIVISOR)
030754  005516                      ADC     @SP         ;  I        ;SET ''CARRY''
030756  060400                      ADD     R4,R0       ;<-
030760  005516            M.DP42:   ADC     @SP                     ;SET ''CARRY''
030762  005716                      TST     @SP                     ;TEST THE UPDATE INDICATOR
030764  001401                      BEQ     .+4         ;->         ;IF ZERO FORGET IT
030766  005203                      INC     R3          ;  I        ;NO CARRY POSSIBLE HERE
030770  005366  000006              DEC     6(SP)       ;<-         ;DECREMENT COUNTER
030774  003347                      BGT     M.DP40                  ;BRANCH IF MORE TO DO
030776  006003                      ROR     R3
031000  103404                      BCS     M.DP44
031002  060501                      ADD     R5,R1
031004  005500                      ADC     R0
031006  060400                      ADD     R4,R0
031010  000241                      CLC
031012  006103            M.DP44:   ROL     R3
031014  062706  000010              ADD     #10,SP                  ;ADJUST STACK BY 4 WORDS
031020  000242                      CLV
031022  000207                      RTS     PC
031024  062706  000006    M.DP50:   ADD     #6,SP
031030  000262                      SEV
031032  000207                      RTS     PC



                          ;ROUTINE TO REPLACE LEADING ZEROS IN A NUMERIC STRING WITH SPACES
                          ;CALL
                          ;         MOV     #ADR,-(SP)              ;ADDRESS OF NUMBER (IN ASCII)
                          ;         JSR     R5,REPLZ
                          ;         .WORD   N                       ;'N' IS NUMBER OF DIGITS TO BE TYPED

031034  010046            REPLZ:    MOV     R0,-(SP)                ;SAVE R0
031036  012746  000012              MOV     #10.,-(SP)              ;MAXIMUM NUMBER OF DIGITS TO BE TYPED
031042  162516                      SUB     (R5)+,(SP)              ;SUBTACT DIGITS TO FORM INDEX.
031044  016600  000006              MOV     6(SP),R0                ;ADDRESS OF NUMBER TO R0
031050  122710  000060    1$:       CMPB    #'0,(R0)                ;BYTE EQUAL TO ASCII '0' ?
031054  001004                      BNE     2$                      ;BR IF NOT
```

```
3346 031056  112710  000040                     MOVB    #40,(R0)          ;REPLACE THE ZERO WITH A SPACE
3347 031062  005200                             INC     R0                ;INCREMENT THE BYTE ADDRESS
3348 031064  000771                             BR      1$                ;GO BACK AND LOOK FOR MORE LEADING ZEROS
3349 031066  105710                     2$:     TSTB    (R0)              ;SEE IF ZERO BYTE TERMINATOR
3350 031070  001003                             BNE     3$                ;BR IF NOT
3351 031072  005300                             DEC     R0                ;BACKUP STRING POINTER
3352 031074  112710  000060                     MOVB    #'0,(R0)          ;PUT A ZERO BACK IN
3353 031100  016637  000006  031114     3$:     MOV     6(SP),4$          ;PUT ADDRESS IN LOCATION FOR TYPEOUT
3354 031106  062637  031114                     ADD     (SP)+,4$          ;BEGINNING OF SIGNIFICANT DIGITS
3355 031112  104401                             TYPE                      ;TYPE THE NUMBER
3356 031114  000000                     4$:     .WORD   0                 ;ADDRESS OF NUMBER
3357 031116  012600                             MOV     (SP)+,R0          ;RESTORE R0
3358 031120  012616                             MOV     (SP)+,(SP)        ;MOVE RETURN ADDRESS
3359 031122  000205                             RTS     R5                ;RETURN
3360
3361                                    ;TYPE NUMERICAL ASCIZ STRING SUPRESS LEADING ZEROS
3362
3363                                    ;CALL
3364                                    ;       MOV     #NUMADR,-(SP)     ;FIRST ADDRESS OF ASCIZ STRING
3365                                    ;       JSR     PC,$SUPRS
3366
3367 031124  010046                     $SUPRS: MOV     R0,-(SP)          ;SAVE R0
3368 031126  016600  000004                     MOV     4(SP),R0          ;PICKUP THE POINTER
3369 031132  105710                     1$:     TSTB    (R0)              ;TERMINATOR ?
3370 031134  001403                             BEQ     2$                ;BR IF YES
3371 031136  122720  000060                     CMPB    #'0,(R0)+         ;IS THIS AN ASCII '0' ?
3372 031142  001773                             BEQ     1$                ;BR IF YES
3373 031144  005300                     2$:     DEC     R0                ;BACKUP BY '1'
3374 031146  010037  031154                     MOV     R0,3$             ;SAVE FOR TYPING
3375 031152  104414                             DISPLY                    ;GO PRINT
3376 031154  000000                     3$:     .WORD   0                 ;ASCIZ POINTER GOES HERE
3377 031156  012600                             MOV     (SP)+,R0          ;RESTORE R0
3378 031160  012616                             MOV     (SP)+,(SP)        ;RESTORE THE STACK
3379 031162  000207                             RTS     PC                ;RETURN
3380
3381                                    ;ROUTINE TO TYPE AT PRIORITY 4
3382
3383 031164  013746  177776             TYPRI4: MOV     @#PS,-(SP)        ;SAVE THE PRESENT STATUS
3384 031170  012737  000200  177776             MOV     #200,@#PS         ;CHANGE THE PRIORITY TO 4
3385 031176  012537  031206                     MOV     (R5)+,1$          ;MESSAGE ADDRESS
3386 031202  004737  032606                     JSR     PC,$TYPE          ;TYPE THE MESSAGE
3387 031206  000000                     1$:     .WORD   0                 ;MESSAGE ADDRESS GOES HERE
3388 031210  000205                             RTS     R5                ;RETURN
3389
3390                                    ;ROUTINE TO TYPE ERRORS
3391                                    ;CALL
3392                                    ;       DISPLY                    ;MUST DEFINED IN 'TRAP' TABLE
3393                                    ;       MESADR                    ;ADDRESS OF MESSAGE
3394                                    ;       RETURN
3395
3396 031212  032777  020000  147734     $DSPLY: BIT     #BIT13,@SWR       ;INHIBIT ERROR TYPEOUT ?
3397 031220  001004                             BNE     1$                ;BR IF YES
3398 031222  005037  177776                     CLR     @#PS              ;SET PRIORITY TO ZERO
3399 031226  000137  032606                     JMP     $TYPE             ;TYPE THE MESSAGE
3400 031232  062716  000002             1$:     ADD     #2,(SP)           ;INCREMENT THE RETURN
3401 031236  000002                             RTI                       ;RETURN
3402
```

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

031240  121127  000060                  CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
031244  103407                                  BLO     1$              ;YES -- BRANCH
031246  121127  000067                          CMPB    (R1),#'7        ;GREATER THAN SEVEN?
031252  101004                                  BHI     1$              ;YES -- BRANCH
031254  111102                                  MOVB    (R1),R2         ;GET THE CHARACTER
031256  042702  177770                          BIC     #^C7,R2         ;STRIP AWAY THE ASCII
031262  005725                                  TST     (R5)+           ;ADJUST FOR RETURN
031264  000205                          1$:     RTS     R5              ;RETURN

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

031266  121127  000060                  CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
031272  103407                                  BLO     1$              ;YES -- BRANCH
031274  121127  000071                          CMPB    (R1),#'9        ;GREATER THAN NINE?
031300  101004                                  BHI     1$              ;YES -- BRANCH
031302  111102                                  MOVB    (R1),R2         ;GET THE CHARACTER
031304  042702  000060                          BIC     #'0,R2          ;STRIP AWAY THE ASCII
031310  005725                                  TST     (R5)+           ;ADJUST FOR RETURN
031312  000205                          1$:     RTS     R5              ;RETURN

                                        ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
                                        ;DETERMINE WHAT IT IS.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R5,CK.CHR       ;CHECK CHARACTER
                                        ;       RETURN  ADR1            ;UNKNOWN CHARACTER
                                        ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
                                        ;       RETURN  ADR3            ;COMMA * (R1)=ADR+1
                                        ;       RETURN  ADR4            ;PERIOD * (R1)=ADR+1
                                        ;       RETURN  ADR5            ;DIGIT BETWEEN 0 AND 7.
                                        ;       RETURN  ADR6            ;DIGIT BETWEEN 8 AND 9.
                                        ;                               ;R2 = DIGIT * (R1)=ADR+1

031314  105711                          CK.CHR: TSTB    (R1)            ;''CARRIAGE RETURN''?
031316  001417                                  BEQ     3$              ;YES -- BRANCH
031320  121127  000054                          CMPB    (R1),#',        ;''COMMA''?
031324  001413                                  BEQ     2$              ;YES -- BRANCH
031326  121127  000056                          CMPB    (R1),#'.        ;''PERIOD''?
031332  001407                                  BEQ     1$              ;YES -- BRANCH
031334  004537  031266                          JSR     R5,CK.DEC       ;''DIGIT''?
031340  000410                                  BR      4$              ;NO -- BRANCH
```

```
031342  004537  031240                  JSR     R5,CK.OCT           ;OCTAL ?
031346  005725                          TST     (R5)+               ;DIGIT BETWEEN 8-9
031350  005725                          TST     (R5)+               ;DIGIT BETWEEN 0-7
031352  005725                  1$:     TST     (R5)+               ;PERIOD
031354  005725                  2$:     TST     (R5)+               ;COMMA
031356  005725                  3$:     TST     (R5)+               ;CARRIAGE RETURN
031360  005201                          INC     R1                  ;MOVE POINTER TO NEXT CHARACTER
031362  011505                  4$:     MOV     (R5),R5             ;UNKNOWN CHARACTER
031364  000205                          RTS     R5                  ;RETURN

                                ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
                                ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
                                ;CALL
                                ;       MOV     #ADR,R1             ;ADDRESS OF ASCIZ STRING
                                ;       MOV     #NUM,R2             ;MAX. MAGNITUDE OF INPUT NUMBER
                                ;       JSR     R5,CK.DIG           ;CHECK DIGITS
                                ;       RETURN  ADR1                ;"CR" ONLY ENTERED -- R2=0
                                ;       RETURN  ADR2                ;"PERIOD" ONLY ENTERED -- R2=0
                                ;       RETURN  ADR3                ;ILLEGAL CHARACTER OR INPUT TOO LARGE -- R2=?
                                ;       RETURN  ADR4                ;"CR" -- R2 = NUMBER
                                ;       RETURN  ADR5                ;"COMMA" -- R2 = NUMBER
                                ;       RETURN  ADR6                ;"PERIOD" -- R2 = NUMBER

031366  010446                  CK.DIG: MOV     R4,-(SP)            ;SAVE R4
031370  010346                          MOV     R3,-(SP)            ;SAVE R3
031372  010246                          MOV     R2,-(SP)            ;SAVE THE MAX. SIZE ON THE STACK
031374  005002                          CLR     R2                  ;START WITH 0
031376  005003                          CLR     R3
031400  005004                          CLR     R4
031402  004537  031314                  JSR     R5,CK.CHR           ;CHECK ONE CHARACTER
031406  031502                          6$                          ;ILLEGAL CHARACTER
031410  031510                          9$                          ;CARRIAGE RETURN
031412  031502                          6$                          ;" "
031414  031504                          7$                          ;","
                                                                    ; .
031416  031422                          1$                          ;DIGIT 0-7
031420  031422                          1$                          ;DIGIT 8-9
031422  062705  000004          1$:     ADD     #4,R5               ;STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
031426  006303                  2$:     ASL     R3                  ;INPUT NUMBER *2
031430  010346                          MOV     R3,-(SP)            ;SAVE *2
031432  006303                          ASL     R3                  ;*4
031434  006303                          ASL     R3                  ;*8
031436  062603                          ADD     (SP)+,R3            ;(*2)+(*8) = *10
031440  060203                          ADD     R2,R3               ;UPDATE THE INPUT NUMBER
031442  004537  031314                  JSR     R5,CK.CHR           ;CHECK ONE CHARACTER
031446  031506                          8$                          ;ILLEGAL CHARACTER
031450  031472                          5$                          ;CARRIAGE RETURN
031452  031470                          4$                          ;" "
031454  031462                          3$                          ;","
                                                                    ; .
031456  031426                          2$                          ;DIGIT 0-7
031460  031426                          2$                          ;DIGIT 8-9
031462  105711                  3$:     TSTB    (R1)                ;DOES A "CR" FOLLOW THE "PERIOD"
031464  001010                          BNE     8$                  ;BR IF NOT
031466  005724                          TST     (R4)+               ;INCREMENT THE RETURN
031470  005724                  4$:     TST     (R4)+               ;INCREMENT THE RETURN
031472  005724                  5$:     TST     (R4)+               ;INCREMENT THE RETURN
031474  020316                          CMP     R3,(SP)             ;CHECK THE MAGNITUDE OF THE NUMBER
031476  101004                          BHI     9$                  ;BR IF ENTERED NUMBER TOO LARGE
```

```
031500  000402                          BR      8$                      ;BYPASS INCREMENT
031502  005725                  6$:     TST     (R5)+                   ;INCREMENT RETURN PAST INVALID RETURN
031504  005725                  7$:     TST     (R5)+                   ;INCREMENT RETURN
031506  060405                  8$:     ADD     R4,R5                   ;SETUP RETURN POINTER
031510  010302                  9$:     MOV     R3,R2                   ;ENTERED VALUE
031512  005726                          TST     (SP)+                   ;CLEAN MAX. SIZE OFF OF STACK
031514  012603                          MOV     (SP)+,R3                ;RESTORE R3
031516  012604                          MOV     (SP)+,R4                ;RESTORE R4
031520  011505                          MOV     (R5),R5                 ;GET RETURN ADDRESS
031522  000205                          RTS     R5                      ;RETURN

                                ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                ;UNSIGNED DECIMAL ASCIZ NUMBER.
                                ;CALL
                                ;       MOV     NUMBER,-(SP)            ;PUT THE NUMBER ON THE STACK
                                ;       JSR     PC,$SB2D                ;CALL
                                ;       RETURN                          ;ADDRESS OF THE 1ST ASCIZ CHAR IS ON THE STACK
                                ;
                                ;NOTE:  THE PROGRAM REQUIRES THIS FORM OF '$SB2D', NOT THE VERSION ON
                                ;       THE SYSMAC LIBRARY, REV C AND LATER

031524  016637  000002  031550  $SB2D:  MOV     2(SP),1$                ;SAVE THE BINARY NUMBER
031532  012746  031550                  MOV     #1$,-(SP)               ;SET THE POINTER
031536  004737  035212                  JSR     PC,$DB2D                ;CALL THE DOUBLE LENGTH CONVERT
031542  012666  000002                  MOV     (SP)+,2(SP)             ;PICKUP THE POINTER
031546  000207                          RTS     PC                      ;RETURN
031550  000000  000000          1$:     .WORD   0,0

                                ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                ;UNSIGNED OCTAL ASCIZ NUMBER.
                                ;CALL
                                ;       MOV     NUMBER,-(SP)            ;PUT THE NUMBER ON THE STACK
                                ;       JSR     PC,$SB2O                ;CALL
                                ;       RETURN                          ;ADDRESS OF THE 1ST ASCIZ CHAR IS ON THE STACK
                                ;
                                ;NOTE:  THE PROGRAM REQUIRES THIS FORM OF '$SB2O', NOT THE VERSION ON
                                ;       THE SYSMAC LIBRARY, REV C AND LATER

031554  016637  000002  031600  $SB2O:  MOV     2(SP),1$                ;SAVE THE BINARY NUMBER
031562  012746  031600                  MOV     #1$,-(SP)               ;SET THE POINTER
031566  004737  035406                  JSR     PC,$DB2O                ;CALL THE DOUBLE LENGTH CONVERT
031572  012666  000002                  MOV     (SP)+,2(SP)             ;PICKUP THE POINTER
031576  000207                          RTS     PC                      ;RETURN
031600  000000  000000          1$:     .WORD   0,0

                                ;KEYBOARD INTERRUPT INITIALIZATION ROUTINE
                                ;CALL
                                ;       JSR     PC,$TKINT
                                ;       RETURN

031604  012737  031634  000060  $TKINT: MOV     #$TKSRV,TKVEC           ;SETUP VECTOR
031612  012737  000100  000062          MOV     #100,TKVEC+2            ;PRIORITY TO 4
031620  005777  147336                  TST     @$TKB                   ;CLEAR THE BUFFER
031624  012777  000100  147326          MOV     #BIT06,@$TKS            ;SET INTERRUPT ENABLE
031632  000207                          RTS     PC                      ;RETURN

                                ;KEYBOARD INTERRUPT SERVICE ROUTINE
                                ;CALL
                                ;               ENTER VIA INTERRUPT

031634  104410                  $TKSRV: RDCHR                   ;READ THE KEYBOARD
031636  112637  031764                  MOVB    (SP)+,5$        ;GET THE CHARACTER
031642  023727  031764  000003          CMP     5$,#3           ;'CONTROL C' ?
031650  001012                          BNE     1$              ;BR IF NOT
031652  104401  001203                  TYPE    ,$CRLF          ;CR-LF
031656  104401  032256                  TYPE    ,$CNTLC         ;'^C'
031662  012737  177777  001362          MOV     #-1,CFLAG       ;SET THE 'CONTROL C' FLAG
031670  005077  147264                  CLR     @$TKS           ;CLEAR THE TTY INTERRUPT
031674  000432                          BR      4$              ;EXIT
031676  023727  001154  000176  1$:     CMP     SWR,#SWREG      ;SOFTWARE SWITCH REGISTER IN USE ?
031704  001024                          BNE     3$              ;BR IF NOT
031706  023727  031764  000007          CMP     5$,#7           ;'CONTROL G' ?
031714  001020                          BNE     3$              ;BR IF NOT
031716  104401  001203                  TYPE    ,$CRLF          ;CR-LF
031722  104401  034765                  TYPE    ,$CNTLG         ;'^G'
031726  013746  177776                  MOV     PS,-(SP)        ;PUT THE STATUS WORD ON THE STACK
031732  012746  031746                  MOV     #2$,-(SP)       ;RETURN ADDRESS
031736  005077  147216                  CLR     @$TKS           ;CLEAR THE TTY INTERRUPT ENABLE
031742  000137  034426                  JMP     $GTSWR          ;GET THE SWITCH REGISTER ENTRY
031746  012777  000100  147204  2$:     MOV     #100,@$TKS      ;ENABLE TTY KEYBOARD INTERRUPT
031754  000402                          BR      4$              ;EXIT
031756  104401  031764          3$:     TYPE    ,5$             ;ECHO THE CHARACTER
031762  000002                  4$:     RTI                     ;RETURN

031764  000000                  5$:     .WORD   0               ;ENTERED CHARACTER

                                ;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;CALL:
                                ;               RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;               RETURN HERE             ;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;                                       ;TERMINATOR WILL BE A BYTE OF ALL 0'S

031766  010346                  $RDLIN: MOV     R3,-(SP)        ;SAVE R3
031770  005046                          CLR     -(SP)           ;CLEAR THE RUBOUT KEY
031772  012703  032244          1$:     MOV     #$TTYIN,R3      ;GET ADDRESS
031776  022703  032256          2$:     CMP     #$TTYIN+10.,R3  ;BUFFER FULL?
032002  101467                          BLOS    4$              ;BR IF YES
032004  104410                          RDCHR                   ;GO READ ONE CHARACTER FROM THE TTY
032006  112613                          MOVB    (SP)+,(R3)      ;GET CHARACTER
032010  122713  000177                  CMPB    #177,(R3)       ;IS IT A RUBOUT
032014  001022                          BNE     5$              ;BR IF NO
032016  005716                          TST     (SP)            ;IS THIS THE FIRST RUBOUT?
032020  001007                          BNE     6$              ;BR IF NO
032022  112737  000134  032242          MOVB    #'\,9$          ;TYPE A BACK SLASH
032030  104401  032242                  TYPE    ,9$
032034  012716  177777                  MOV     #-1,(SP)        ;SET THE RUBOUT KEY
032040  005303                  6$:     DEC     R3              ;BACKUP BY ONE
032042  020327  032244                  CMP     R3,#$TTYIN      ;STACK EMPTY?
032046  103445                          BLO     4$              ;BR IF YES
032050  111337  032242                  MOVB    (R3),9$         ;SETUP TO TYPEOUT THE DELETED CHAR.
032054  104401  032242                  TYPE    ,9$             ;GO TYPE
032060  000746                          BR      2$              ;GO READ ANOTHER CHAR.
032062  005716                  5$:     TST     (SP)            ;RUBOUT KEY SET?
032064  001406                          BEQ     7$              ;BR IF NO
```

```
032066  112737  000134  032242          MOVB    #'\,9$                  ;TYPE A BACK SLASH
032074  104401  032242                  TYPE    ,9$
032100  005016                          CLR     (SP)                    ;CLEAR THE RUBOUT KEY
032102  122713  000025          7$:     CMPB    #25,(R3)                ;IS CHARACTER A CTRL U?
032106  001003                          BNE     10$                     ;BR IF NO
032110  104401  034760                  TYPE    ,$CNTLU                 ;TYPE A CONTROL 'U''
032114  000726                          BR      1$                      ;GO START OVER
032116  122713  000003          10$:    CMPB    #3,(R3)                 ;IS CHARACTER A CTRL C ?
032122  001006                          BNE     8$                      ;BR IF NOT
032124  012737  177777  001362          MOV     #-1,CFLAG               ;SET CNTRL C FLAG
032132  104401  032256                  TYPE    ,$CNTLC                 ;ECHO IT
032136  000427                          BR      11$                     ;EXIT
032140  122713  000012          8$:     CMPB    #12,(R3)                ;IS CHARACTER A 'LF''?
032144  001011                          BNE     3$                      ;BRANCH IF NO
032146  105013                          CLRB    (R3)                    ;CLEAR THE CHARACTER
032150  104401  001203                  TYPE    ,$CRLF                  ;TYPE A ''CR'' & 'LF''
032154  104401  032244                  TYPE    ,$TTYIN                 ;TYPE THE INPUT STRING
032160  000706                          BR      2$                      ;GO PICKUP ANOTHER CHACTER
032162  104401  001202          4$:     TYPE    ,$QUES                  ;TYPE A '?'
032166  000701                          BR      1$                      ;CLEAR THE BUFFER AND LOOP
032170  111337  032242          3$:     MOVB    (R3),9$                 ;ECHO THE CHARACTER
032174  104401  032242                  TYPE    ,9$
032200  122723  000015                  CMPB    #15,(R3)+               ;CHECK FOR RETURN
032204  001274                          BNE     2$                      ;LOOP IF NOT RETURN
032206  105063  177777                  CLRB    -1(R3)                  ;CLEAR RETURN (THE 15)
032212  104401  001204                  TYPE    ,$LF                    ;TYPE A LINE FEED
032216  005726                  11$:    TST     (SP)+                   ;CLEAN RUBOUT KEY FROM THE STACK
032220  012603                          MOV     (SP)+,R3                ;RESTORE R3
032222  011646                          MOV     (SP),-(SP)              ;ADJUST THE STACK AND PUT ADDRESS OF THE
032224  016666  000004  000002          MOV     4(SP),2(SP)             ;FIRST ASCII CHARACTER ON IT
032232  012766  032244  000004          MOV     #$TTYIN,4(SP)
032240  000002                          RTI                             ;RETURN
032242     000                  9$:     .BYTE   0                       ;STORAGE FOR ASCII CHAR. TO TYPE
032243     000                          .BYTE   0                       ;TERMINATOR
032244                          $TTYIN: .BLKB   10.                     ;RESERVE 10 BYTES FOR TTY INPUT
032256     136     103     200  $CNTLC: .ASCIZ  /^C/<CRLF>              ;CONTROL ''C''


                                .EVEN

                                .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 $ERRTYP 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
                                ;;*CALL
                                ;;*        ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

        032262                  $ERROR:
        032262  104407                  CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
        032264  010337  001344          MOV     R3,ATTN                 ;SAVE THE ATTENTION REGISTER CONTENTS
```

```
032270  010137  001220                    MOV     R1,DRIVE          ;DRIVE NUMBER
032274  032777  020000  146652            BIT     #SW13,@SWR        ;INHIBIT PRINTOUTS ?
032302  001004                            BNE     .+12              ;BR IF YES
032304  104401  001203                    TYPE    ,$CRLF            ;CR-LF
032310  004737  024364                    JSR     PC,$TIME          ;TYPE THE TIME
032314  105237  001117            7$:     INCB    $ERFLG            ;;SET THE ERROR FLAG
032320  001775                            BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
032322  013777  001116  146626            MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
032330  032777  002000  146616            BIT     #BIT10,@SWR       ;;BELL ON ERROR?
032336  001402                            BEQ     1$                ;;NO - SKIP
032340  104401  001176                    TYPE    ,$BELL            ;;RING BELL
032344  005237  001126            1$:     INC     $ERTTL            ;;COUNT THE NUMBER OF ERRORS
032350  011637  001132                    MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
032354  162737  000002  001132            SUB     #2,$ERRPC
032362  117737  146544  001130            MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
032370  032777  020000  146556            BIT     #BIT13,@SWR       ;;SKIP TYPEOUT IF SET
032376  001004                            BNE     20$               ;;SKIP TYPEOUTS
032400  004737  032452                    JSR     PC,$ERRTYP        ;;GO TO USER ERROR ROUTINE
032404  104401  001203                    TYPE    ,$CRLF
032410                            20$:
032410  122737  000001  001226            CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
032416  001007                            BNE     2$                ;;NO,SKIP APT ERROR REPORT
032420  113737  001130  032432            MOVB    $ITEMB,21$        ;;SET ITEM NUMBER AS ERROR NUMBER
032426  004737  033160                    JSR     FC,$ATY4          ;;REPORT FATAL ERROR TO APT
032432     000                    21$:    .BYTE   0
032433     000                            .BYTE   0
032434  000777                    22$:    BR      22$               ;;APT ERROR LOOP
032436  005777  146512            2$:     TST     @SWR              ;;HALT ON ERROR
032442  100002                            BPL     3$                ;;SKIP IF CONTINUE
032444  000000                            HALT                      ;;HALT ON ERROR!
032446  104407                            CKSWR                     ;;TEST FOR CHANGE IN SOFT-SWR
032450                            3$:
032450  000002                            RTI                       ;;RETURN

                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                  ;;***********************************************************************
                                  ;;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
                                  ;;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE ''ERROR TABLE'' ($ERRTB),
                                  ;;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

032452                            $ERRTYP:
032452  104401  001203                    TYPE    ,$CRLF            ;;''CARRIAGE RETURN'' & ''LINE FEED''
032456  010046                            MOV     R0,-(SP)          ;;SAVE R0
032460  005000                            CLR     R0                ;;PICKUP THE ITEM INDEX
032462  153700  001130                    BISB    @#$ITEMB,R0
032466  001004                            BNE     1$                ;;IF ITEM NUMBER IS ZERO, JUST
                                                                    ;;TYPE THE PC OF THE ERROR
032470  013746  001132                    MOV     $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
                                                                    ;;ERROR ADDRESS
032474  104402                            TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032476  000426                            BR      6$                ;;GET OUT
032500  005300                    1$:     DEC     R0                ;;ADJUST THE INDEX SO THAT IT WILL
032502  006300                            ASL     R0                ;;      WORK FOR THE ERROR TABLE
032504  006300                            ASL     R0
032506  006300                            ASL     R0
032510  062700  003546                    ADD     #$ERRTB,R0        ;;FORM TABLE POINTER
```

```
032514  012037  032524                  MOV     (R0)+,2$                ;;PICKUP ''ERROR MESSAGE'' POINTER
032520  001404                          BEQ     3$                      ;;SKIP TYPEOUT IF NO POINTER
032522  104401                          TYPE                            ;;TYPE THE ''ERROR MESSAGE''
032524  000000                  2$:     .WORD   0                       ;;''ERROR MESSAGE'' POINTER GOES HERE
032526  104401  001203                  TYPE    ,$CRLF                  ;;''CARRIAGE RETURN'' & ''LINE FEED''
032532  012037  032542          3$:     MOV     (R0)+,4$                ;;PICKUP ''DATA HEADER'' POINTER
032536  001404                          BEQ     5$                      ;;SKIP TYPEOUT IF 0
032540  104401                          TYPE                            ;;TYPE THE ''DATA HEADER''
032542  000000                  4$:     .WORD   0                       ;;''DATA HEADER'' POINTER GOES HERE
032544  104401  001203                  TYPE    ,$CRLF                  ;;''CARRIAGE RETURN'' & ''LINE FEED''
032550  011000                  5$:     MOV     (R0),R0                 ;;PICKUP ''DATA TABLE'' POINTER
032552  001004                          BNE     7$                      ;;GO TYPE THE DATA
032554  012600                  6$:     MOV     (SP)+,R0                ;;RESTORE R0
032556  104401  001203                  TYPE    ,$CRLF                  ;;''CARRIAGE RETURN'' & ''LINE FEED''
032562  000207                          RTS     PC                      ;;RETURN
032564                          7$:
032564  013046                          MOV     @(R0)+,-(SP)            ;;SAVE @(R0)+ FOR TYPEOUT
032566  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032570  005710                          TST     (R0)                    ;;IS THERE ANOTHER NUMBER?
032572  001770                          BEQ     6$                      ;;BR IF NO
032574  104401  032602                  TYPE    ,8$                     ;;TYPE TWO(2) SPACES
032600  000771                          BR      7$                      ;;LOOP
032602     040     040     000  8$:     .ASCIZ  /  /                    ;;TWO(2) SPACES
                                        .EVEN

                                .SBTTL  TYPE ROUTINE

                                ;;*******************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR                 ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

032606  105737  001173          $TYPE:  TSTB    $TPFLG                  ;;IS THERE A TERMINAL?
032612  100002                          BPL     1$                      ;;BR IF YES
032614  000000                          HALT                            ;;HALT HERE IF NO TERMINAL
032616  000430                          BR      3$                      ;;LEAVE
032620  010046                  1$:     MOV     R0,-(SP)                ;;SAVE R0
032622  017600  000002                  MOV     @2(SP),R0               ;;GET ADDRESS OF ASCIZ STRING
032626  122737  000001  001226          CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
032634  001011                          BNE     62$                     ;;NO,GO CHECK FOR APT CONSOLE
032636  132737  000100  001227          BITB    #APTSPOOL,$ENVM         ;;SPOOL MESSAGE TO APT
032644  001405                          BEQ     62$                     ;;NO,GO CHECK FOR CONSOLE
032646  010037  032656                  MOV     R0,61$                  ;;SETUP MESSAGE ADDRESS FOR APT
032652  004737  033150                  JSR     PC,$ATY3                ;;SPOOL MESSAGE TO APT
032656  000000                  61$:    .WORD   0                       ;;MESSAGE ADDRESS
032660  132737  000040  001227  62$:    BITB    #APTCSUP,$ENVM          ;;APT CONSOLE SUPPRESSED
032666  001003                          BNE     60$                     ;;YES,SKIP TYPE OUT
```

```
032670  112046                  2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
032672  001005                          BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
032674  005726                          TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
032676  012600                  60$:    MOV     (SP)+,R0                ;;RESTORE R0
032700  062716  000002          3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
032704  000002                          RTI                             ;;RETURN
032706  122716  000011          4$:     CMPB    #HT,(SP)                ;;BRANCH IF <HT>
032712  001430                          BEQ     8$
032714  122716  000200                  CMPB    #CRLF,(SP)              ;;BRANCH IF NOT <CRLF>
032720  001006                          BNE     5$
032722  005726                          TST     (SP)+                   ;;POP  <CR><LF> EQUIV
032724  104401                          TYPE                            ;;TYPE A CR AND LF
032726  001203                          $CRLF
032730  105037  033136                  CLRB    $CHARCNT                ;;CLEAR CHARACTER COUNT
032734  000755                          BR      2$                      ;;GET NEXT CHARACTER
032736  004737  033020          5$:     JSR     PC,$TYPEC               ;;GO TYPE THIS CHARACTER
032742  123726  001172          6$:     CMPB    $FILLC,(SP)+            ;;IS IT TIME FOR FILLER CHARS.?
032746  001350                          BNE     2$                      ;;IF NO GO GET NEXT CHAR.
032750  013746  001170                  MOV     $NULL,-(SP)             ;;GET # OF FILLER CHARS. NEEDED
                                                                        ;;AND THE NULL CHAR.
032754  105366  000001          7$:     DECB    1(SP)                   ;;DOES A NULL NEED TO BE TYPED?
032760  002770                          BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
032762  004737  033020                  JSR     PC,$TYPEC               ;;GO TYPE A NULL
032766  105337  033136                  DECB    $CHARCNT                ;;DO NOT COUNT AS A COUNT
032772  000770                          BR      7$                      ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

032774  112716  000040          8$:     MOVB    #' ,(SP)                ;;REPLACE TAB WITH SPACE
033000  004737  033020          9$:     JSR     PC,$TYPEC               ;;TYPE A SPACE
033004  132737  000007  033136          BITB    #7,$CHARCNT             ;;BRANCH IF NOT AT
033012  001372                          BNE     9$                      ;;TAB STOP
033014  005726                          TST     (SP)+                   ;;POP SPACE OFF STACK
033016  000724                          BR      2$                      ;;GET NEXT CHARACTER
033020                          $TYPEC:
033020  105777  146134                  TSTB    @$TKS                   ;;CHAR IN KYBD BUFFER?
033024  100022                          BPL     10$                     ;;BR IF NOT
033026  017746  146130                  MOV     @$TKB,-(SP)             ;;GET CHAR
033032  042716  177600                  BIC     #177600,(SP)            ;;STRIP EXTRANEOUS BITS
033036  122716  000023                  CMPB    #$XOFF,(SP)             ;;WAS CHAR XOFF
033042  001012                          BNE     102$                    ;;BR IF NOT
033044                          101$:
033044  105777  146110                  TSTB    @$TKS                   ;;WAIT FOR CHAR
033050  100375                          BPL     101$
033052  117716  146104                  MOVB    @$TKB,(SP)              ;;GET CHAR
033056  042716  177600                  BIC     #177600,(SP)            ;;STRIP IT
033062  122716  000021                  CMPB    #$XON,(SP)              ;;WAS IT XON?
033066  001366                          BNE     101$                    ;;BR IF NOT
033070                          102$:
033070  005726                          TST     (SP)+                   ;;FIX STACK
033072                          10$:
033072  105777  146066                  TSTB    @$TPS                   ;;WAIT UNTIL PRINTER IS READY
033076  100375                          BPL     10$
033100  116677  000002  146060          MOVB    2(SP),@$TPB             ;;LOAD CHAR TO BE TYPED INTO DATA REG.
033106  122766  000015  000002          CMPB    #CR,2(SP)               ;;IS CHARACTER A CARRIAGE RETURN?
033114  001003                          BNE     1$                      ;;BRANCH IF NO
033116  105037  033136                  CLRB    $CHARCNT                ;;YES--CLEAR CHARACTER COUNT
```

```
033122  000406                                  BR      $TYPEX          ;;EXIT
033124  122766  000012  000002  1$:             CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
033132  001402                                  BEQ     $TYPEX          ;;BRANCH IF YES
033134  105227                                  INCB    (PC)+           ;;COUNT THE CHARACTER
033136  000000                  $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
033140  000207                  $TYPEX: RTS     PC

                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
033142  112737  000001  033406  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
033150  112737  000001  033404  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
033156  000403                          BR      $ATYC
033160  112737  000001  033406  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
033166                          $ATYC:
033166  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033170  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033172  105737  033404                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
033176  001450                          BEQ     5$              ;;IF NOT:  BR
033200  122737  000001  001226          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
033206  001031                          BNE     3$              ;;IF NOT:  BR
033210  132737  000100  001227          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
033216  001425                          BEQ     3$              ;;IF NOT:  BR
033220  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
033224  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
033232  005737  001206          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
033236  001375                          BNE     1$              ;;IF NOT:  WAIT
033240  010037  001222                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
033244  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
033246  001376                          BNE     2$
033250  163700  001222                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
033254  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
033256  010037  001224                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
033262  012737  000004  001206          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
033270  000413                          BR      5$
033272  017637  000004  033316  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
033300  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
033306  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
033312  004737  032606                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
033316  000000                  4$:     .WORD   0
033320                          5$:
033320  105737  033406          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
033324  001416                          BEQ     12$             ;;IF NOT:  BR
033326  005737  001226                  TST     $ENV            ;;RUNNING UNDER APT?
033332  001413                          BEQ     12$             ;;IF NOT:  BR
033334  005737  001206          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
033340  001375                          BNE     11$             ;;IF NOT:  WAIT
033342  017637  000004  001210          MOV     @4(SP),$FATAL   ;;GET ERROR #
033350  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
033356  005237  001206                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
033362  105037  033406          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
033366  105037  033405                  CLRB    $LFLG           ;;CLEAR LOG FLAG
033372  105037  033404                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
033376  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
033400  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
033402  000207                          RTS     PC              ;;RETURN
```

```
033404     000                    $MFLG:  .BYTE    0                   ;;MESSG. FLAG
033405     000                    $LFLG:  .BYTE    0                   ;;LOG FLAG
033406     000                    $FFLG:  .BYTE    0                   ;;FATAL FLAG
                                          .EVEN
        000200                    APTSIZE = 200
        000001                    APTENV  = 001
        000100                    APTSPOOL= 100
        000040                    APTCSUP = 040

                                  .SBTTL  POWER DOWN AND UP ROUTINES

                                  ;;*****************************************************************
                                  ;POWER DOWN ROUTINE
033410  012737  033554  000024    $PWRDN: MOV      #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
033416  012737  000340  000026            MOV      #340,@#PWRVEC+2  ;;PRIO:7
033424  010046                            MOV      R0,-(SP)         ;;PUSH R0 ON STACK
033426  010146                            MOV      R1,-(SP)         ;;PUSH R1 ON STACK
033430  010246                            MOV      R2,-(SP)         ;;PUSH R2 ON STACK
033432  010346                            MOV      R3,-(SP)         ;;PUSH R3 ON STACK
033434  010446                            MOV      R4,-(SP)         ;;PUSH R4 ON STACK
033436  010546                            MOV      R5,-(SP)         ;;PUSH R5 ON STACK
033440  017746  145510                    MOV      @SWR,-(SP)       ;;PUSH @SWR ON STACK
033444  010637  033560                    MOV      SP,$SAVR6        ;;SAVE SP
033450  012737  033462  000024            MOV      #$PWRUP,@#PWRVEC ;;SET UP VECTOR
033456  000000                            HALT
033460  000776                            BR       .-2              ;;HANG UP

                                  ;;*****************************************************************
                                  ;POWER UP ROUTINE
033462  012737  033554  000024    $PWRUP: MOV      #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
033470  013706  033560                    MOV      $SAVR6,SP        ;;GET SP
033474  005037  033560                    CLR      $SAVR6           ;;WAIT LOOP FOR THE TTY
033500  005237  033560            1$:     INC      $SAVR6           ;;WAIT FOR THE INC
033504  001375                            BNE      1$               ;;OF  WORD
033506  012677  145442                    MOV      (SP)+,@SWR       ;;POP STACK INTO @SWR
033512  012605                            MOV      (SP)+,R5         ;;POP STACK INTO R5
033514  012604                            MOV      (SP)+,R4         ;;POP STACK INTO R4
033516  012603                            MOV      (SP)+,R3         ;;POP STACK INTO R3
033520  012602                            MOV      (SP)+,R2         ;;POP STACK INTO R2
033522  012601                            MOV      (SP)+,R1         ;;POP STACK INTO R1
033524  012600                            MOV      (SP)+,R0         ;;POP STACK INTO R0
033526  012737  033410  000024            MOV      #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
033534  012737  000340  000026            MOV      #340,@#PWRVEC+2  ;;PRIO:7
033542  104401                            TYPE                      ;REPORT THE POWER FAILURE
033544  033562                    $PWRMG: .WORD    $POWER           ;;POWER FAIL MESSAGE POINTER
033546  012716                            MOV      (PC)+,(SP)       ;;RESTART AT SATPOW
033550  033572                    $PWRAD: .WORD    SATPOW           ;;RESTART ADDRESS
033552  000002                            RTI
033554  000000                    $ILLUP: HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
033556  000776                            BR       .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
033560  000000                    $SAVR6: 0                         ;;PUT THE SP HERE
033562     015     012     120    $POWER: .ASCIZ   <15><12>'POWER'
                                          .EVEN


                                                                    ;POW UP ROUTINE , WAIT
                                                                    ;FIVE MINUTS, THEN AUTO STARTS AT SETVEC

```

```
033572  000005                          SATPOW: RESET                           ;CLEAR THE BUS
033574  005037  177776                          CLR     @#PS                    ;CLEAR PSW
033600  005037  001366                          CLR     HOUR                    ;RESET THE HOUR COUNT
033604  005037  001370                          CLR     MINUTE                  ;RESET THE MINUTS COUNT
033610  005037  001372                          CLR     SECOND                  ;RESET THE SECOND COUNT
033614  005037  001474                          CLR     INTRVL+2                ;RESET THE INTERVAL COUNT
033620  004737  023326                          JSR     PC,CKCLK                ;CHECK THE CLOCK
033624  022737  000170  001474          1$:     CMP     #120.,INTRVL+2          ;FIVE MINUTES YET ?
033632  101374                                  BHI     1$                      ;WAIT IF NOT
033634  005037  001474                          CLR     INTRVL+2                ;RESET INTERVAL COUNT
033640  005037  001372                          CLR     SECOND                  ;RESET TIMER
033644  005037  001370                          CLR     MINUTE                  ;
033650  012737  000400  001360                  MOV     #400,CHGADR             ;FORGE THE AUTO START
033656  012705  001526                          MOV     #ORDERQ,R5              ;CLEAR UP THE QUEUE AND BUFFER
033662  005025                          2$:     CLR     (R5)+
033664  022705  002106                          CMP     #BLKADR,R5              ;ALL DONE ?
033670  001374                                  BNE     2$                      ;BRANCH IF NOT
033672  012737  007070  001266                  MOV     #7070,$CDW1             ;FLAG FROM POWER FAIL
033700  000137  005054                          JMP     SIZMEM                  ;LOOP BACK

                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;*****************************************************************
                                        ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;;*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

033704  017646  000000                  $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
033710  116637  000001  034127                  MOVB    1(SP),$0FILL            ;;LOAD ZERO FILL SWITCH
033716  112637  034131                          MOVB    (SP)+,$OMODE+1          ;;NUMBER OF DIGITS TO TYPE
033722  062716  000002                          ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
033726  000406                                  BR      $TYPON
033730  112737  000001  034127          $TYPOC: MOVB    #1,$0FILL               ;;SET THE ZERO FILL SWITCH
033736  112737  000006  034131                  MOVB    #6,$OMODE+1             ;;SET FOR SIX(6) DIGITS
033744  112737  000005  034126          $TYPON: MOVB    #5,$OCNT                ;;SET THE ITERATION COUNT
033752  010346                                  MOV     R3,-(SP)                ;;SAVE R3
033754  010446                                  MOV     R4,-(SP)                ;;SAVE R4
033756  010546                                  MOV     R5,-(SP)                ;;SAVE R5
033760  113704  034131                          MOVB    $OMODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
```

```
     033764  005404                           NEG     R4
     033766  062704  000006                   ADD     #6,R4                  ;;SUBTRACT IT FOR MAX. ALLOWED
     033772  110437  034130                   MOVB    R4,$OMODE              ;;SAVE IT FOR USE
     033776  113704  034127                   MOVB    $OFILL,R4              ;;GET THE ZERO FILL SWITCH
     034002  016605  000012                   MOV     12(SP),R5              ;;PICKUP THE INPUT NUMBER
     034006  005003                           CLR     R3                     ;;CLEAR THE OUTPUT WORD
     034010  006105                   1$:     ROL     R5                     ;;ROTATE MSB INTO ''C''
     034012  000404                           BR      3$                     ;;GO DO MSB
     034014  006105                   2$:     ROL     R5                     ;;FORM THIS DIGIT
     034016  006105                           ROL     R5
     034020  006105                           ROL     R5
     034022  010503                           MOV     R5,R3
     034024  006103                   3$:     ROL     R3                     ;;GET LSB OF THIS DIGIT
     034026  105337  034130                   DECB    $OMODE                 ;;TYPE THIS DIGIT?
     034032  100016                           BPL     7$                     ;;BR IF NO
     034034  042703  177770                   BIC     #177770,R3             ;;GET RID OF JUNK
     034040  001002                           BNE     4$                     ;;TEST FOR 0
     034042  005704                           TST     R4                     ;;SUPPRESS THIS 0?
     034044  001403                           BEQ     5$                     ;;BR IF YES
     034046  005204                   4$:     INC     R4                     ;;DON'T SUPPRESS ANYMORE 0'S
     034050  052703  000060                   BIS     #'0,R3                 ;;MAKE THIS DIGIT ASCII
     034054  052703  000040           5$:     BIS     #' ,R3                 ;;MAKE ASCII IF NOT ALREADY
     034060  110337  034124                   MOVB    R3,8$                  ;;SAVE FOR TYPING
     034064  104401  034124                   TYPE    ,8$                    ;;GO TYPE THIS DIGIT
     034070  105337  034126           7$:     DECB    $OCNT                  ;;COUNT BY 1
     034074  003347                           BGT     2$                     ;;BR IF MORE TO DO
     034076  002402                           BLT     6$                     ;;BR IF DONE
     034100  005204                           INC     R4                     ;;INSURE LAST DIGIT ISN'T A BLANK
     034102  000744                           BR      2$                     ;;GO DO THE LAST DIGIT
     034104  012605                   6$:     MOV     (SP)+,R5               ;;RESTORE R5
     034106  012604                           MOV     (SP)+,R4               ;;RESTORE R4
     034110  012603                           MOV     (SP)+,R3               ;;RESTORE R3
     034112  016666  000002  000004           MOV     2(SP),4(SP)            ;;SET THE STACK FOR RETURNING
     034120  012616                           MOV     (SP)+,(SP)
     034122  000002                           RTI                            ;;RETURN
     034124     000                   8$:     .BYTE   0                      ;;STORAGE FOR ASCII DIGIT
     034125     000                           .BYTE   0                      ;;TERMINATOR FOR TYPE ROUTINE
     034126     000                   $OCNT:  .BYTE   0                      ;;OCTAL DIGIT COUNTER
     034127     000                   $OFILL: .BYTE   0                      ;;ZERO FILL SWITCH
     034130  000000                   $OMODE: .WORD   0                      ;;NUMBER OF DIGITS TO TYPE

                                      .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                      ;;*******************************************************************
                                      ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                      ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                      ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                      ;;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                      ;;*REPLACED WITH SPACES.
                                      ;;*CALL:
                                      ;;*       MOV     NUM,-(SP)             ;;PUT THE BINARY NUMBER ON THE STACK
                                      ;;*       TYPDS                         ;;GO TO THE ROUTINE

     034132                           $TYPDS:
     034132  010046                           MOV     R0,-(SP)              ;;PUSH R0 ON STACK
     034134  010146                           MOV     R1,-(SP)              ;;PUSH R1 ON STACK
     034136  010246                           MOV     R2,-(SP)              ;;PUSH R2 ON STACK
```

```
  034140  010346                          MOV     R3,-(SP)           ;;PUSH R3 ON STACK
  034142  010546                          MOV     R5,-(SP)           ;;PUSH R5 ON STACK
  034144  012746  020200                  MOV     #20200,-(SP)       ;;SET BLANK SWITCH AND SIGN
  034150  016605  000020                  MOV     20(SP),R5          ;;GET THE INPUT NUMBER
  034154  100004                          BPL     1$                 ;;BR IF INPUT IS POS.
  034156  005405                          NEG     R5                 ;;MAKE THE BINARY NUMBER POS.
  034160  112766  000055  000001          MOVB    #'-,1(SP)          ;;MAKE THE ASCII NUMBER NEG.
  034166  005000                  1$:     CLR     R0                 ;;ZERO THE CONSTANTS INDEX
  034170  012703  034346                  MOV     #$DBLK,R3          ;;SETUP THE OUTPUT POINTER
  034174  112723  000040                  MOVB    #' ,(R3)+          ;;SET THE FIRST CHARACTER TO A BLANK
  034200  005002                  2$:     CLR     R2                 ;;CLEAR THE BCD NUMBER
  034202  016001  034336                  MOV     $DTBL(R0),R1       ;;GET THE CONSTANT
  034206  160105                  3$:     SUB     R1,R5              ;;FORM THIS BCD DIGIT
  034210  002402                          BLT     4$                 ;;BR IF DONE
  034212  005202                          INC     R2                 ;;INCREASE THE BCD DIGIT BY 1
  034214  000774                          BR      3$
  034216  060105                  4$:     ADD     R1,R5              ;;ADD BACK THE CONSTANT
  034220  005702                          TST     R2                 ;;CHECK IF BCD DIGIT=0
  034222  001002                          BNE     5$                 ;;FALL THROUGH IF 0
  034224  105716                          TSTB    (SP)               ;;STILL DOING LEADING 0'S?
  034226  100407                          BMI     7$                 ;;BR IF YES
  034230  106316                  5$:     ASLB    (SP)               ;;MSD?
  034232  103003                          BCC     6$                 ;;BR IF NO
  034234  116663  000001  177777          MOVB    1(SP),-1(R3)       ;;YES--SET THE SIGN
  034242  052702  000060          6$:     BIS     #'0,R2             ;;MAKE THE BCD DIGIT ASCII
  034246  052702  000040          7$:     BIS     #' ,R2             ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
  034252  110223                          MOVB    R2,(R3)+           ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
  034254  005720                          TST     (R0)+              ;;JUST INCREMENTING
  034256  020027  000010                  CMP     R0,#10             ;;CHECK THE TABLE INDEX
  034262  002746                          BLT     2$                 ;;GO DO THE NEXT DIGIT
  034264  003002                          BGT     8$                 ;;GO TO EXIT
  034266  010502                          MOV     R5,R2              ;;GET THE LSD
  034270  000764                          BR      6$                 ;;GO CHANGE TO ASCII
  034272  105726                  8$:     TSTB    (SP)+              ;;WAS THE LSD THE FIRST NON-ZERO?
  034274  100003                          BPL     9$                 ;;BR IF NO
  034276  116663  177777  177776          MOVB    -1(SP),-2(R3)      ;;YES--SET THE SIGN FOR TYPING
  034304  105013                  9$:     CLRB    (R3)               ;;SET THE TERMINATOR
  034306  012605                          MOV     (SP)+,R5           ;;POP STACK INTO R5
  034310  012603                          MOV     (SP)+,R3           ;;POP STACK INTO R3
  034312  012602                          MOV     (SP)+,R2           ;;POP STACK INTO R2
  034314  012601                          MOV     (SP)+,R1           ;;POP STACK INTO R1
  034316  012600                          MOV     (SP)+,R0           ;;POP STACK INTO R0
  034320  104401  034346                  TYPE    ,$DBLK             ;;NOW TYPE THE NUMBER
  034324  016666  000002  000004          MOV     2(SP),4(SP)        ;;ADJUST THE STACK
  034332  012616                          MOV     (SP)+,(SP)
  034334  000002                          RTI                        ;;RETURN TO USER
  034336  023420                  $DTBL:  10000.
  034340  001750                          1000.
  034342  000144                          100.
  034344  000012                          10.
  034346                          $DBLK:  .BLKW   4

                                  .SBTTL  TTY INPUT ROUTINE

                                  ;;*************************************************************
                                  .ENABL  LSB
```

```
                                        ;;*******************************************************************
                                        ;;*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 FLAG MODE.
034356  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR              ;;IS THE SOFT-SWR SELECTED?
034364  001074                          BNE     15$                     ;;BRANCH IF NO
034366  105777  144566                  TSTB    @$TKS                   ;;CHAR THERE?
034372  100071                          BPL     15$                     ;;IF NO, DON'T WAIT AROUND
034374  117746  144562                  MOVB    @$TKB,-(SP)             ;;SAVE THE CHAR
034400  042716  177600                  BIC     #^C177,(SP)             ;;STRIP-OFF THE ASCII
034404  022726  000007                  CMP     #7,(SP)+                ;;IS IT A CONTROL G?
034410  001062                          BNE     15$                     ;;NO, RETURN TO USER
034412  123727  001150  000001          CMPB    $AUTOB,#1               ;;ARE WE RUNNING IN AUTO-MODE?
034420  001456                          BEQ     15$                     ;;BRANCH IF YES

034422  104401  034765                  TYPE    ,$CNTLG                 ;;ECHO THE CONTROL-G (^G)
034426  104401  034772          $GTSWR: TYPE    ,$MSWR                  ;;TYPE CURRENT CONTENTS
034432  013746  000176                  MOV     SWREG,-(SP)             ;;SAVE SWREG FOR TYPEOUT
034436  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034440  104401  035003                  TYPE    ,$MNEW                  ;;PROMPT FOR NEW SWR
034444  005046                  19$:    CLR     -(SP)                   ;;CLEAR COUNTER
034446  005046                          CLR     -(SP)                   ;;THE NEW SWR
034450  105777  144504          7$:     TSTB    @$TKS                   ;;CHAR THERE?
034454  100375                          BPL     7$                      ;;IF NOT TRY AGAIN

034456  117746  144500                  MOVB    @$TKB,-(SP)             ;;PICK UP CHAR
034462  042716  177600                  BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII



034466  021627  000025          9$:     CMP     (SP),#25                ;;IS IT A CONTROL-U?
034472  001005                          BNE     10$                     ;;BRANCH IF NOT
034474  104401  034760                  TYPE    ,$CNTLU                 ;;YES, ECHO CONTROL-U (^U)
034500  062706  000006          20$:    ADD     #6,SP                   ;;IGNORE PREVIOUS INPUT
034504  000757                          BR      19$                     ;;LET'S TRY IT AGAIN

034506  021627  000015          10$:    CMP     (SP),#15                ;;IS IT A <CR>?
034512  001022                          BNE     16$                     ;;BRANCH IF NO
034514  005766  000004                  TST     4(SP)                   ;;YES, IS IT THE FIRST CHAR?
034520  001403                          BEQ     11$                     ;;BRANCH IF YES
034522  016677  000002  144424          MOV     2(SP),@SWR              ;;SAVE NEW SWR
034530  062706  000006          11$:    ADD     #6,SP                   ;;CLEAR UP STACK
034534  104401  001203          14$:    TYPE    ,$CRLF                  ;;ECHO <CR> AND <LF>
034540  123727  001151  000001          CMPB    $INTAG,#1               ;;RE-ENABLE TTY KBD INTERRUPTS?
034546  001003                          BNE     15$                     ;;BRANCH IF NOT
034550  012777  000100  144402          MOV     #100,@$TKS              ;;RE-ENABLE TTY KBD INTERRUPTS
034556  000002                  15$:    RTI                             ;;RETURN
034560  004737  033020          16$:    JSR     PC,$TYPEC               ;;ECHO CHAR
034564  021627  000060                  CMP     (SP),#60                ;;CHAR < 0?
034570  002420                          BLT     18$                     ;;BRANCH IF YES
034572  021627  000067                  CMP     (SP),#67                ;;CHAR > 7?
034576  003015                          BGT     18$                     ;;BRANCH IF YES
034600  042726  000060                  BIC     #60,(SP)+               ;;STRIP-OFF ASCII
034604  005766  000002                  TST     2(SP)                   ;;IS THIS THE FIRST CHAR
034610  001403                          BEQ     17$                     ;;BRANCH IF YES
```

```
        034612  006316                                  ASL     (SP)                    ;;NO, SHIFT PRESENT
        034614  006316                                  ASL     (SP)                    ;;    CHAR OVER TO MAKE
        034616  006316                                  ASL     (SP)                    ;;    ROOM FOR NEW ONE.
        034620  005266  000002                  17$:    INC     2(SP)                   ;;KEEP COUNT OF CHAR
        034624  056616  177776                          BIS     -2(SP),(SP)             ;;SET IN NEW CHAR
        034630  000707                                  BR      7$                      ;;GET THE NEXT ONE
        034632  104401  001202                  18$:    TYPE    ,$QUES                  ;;TYPE ?<CR><LF>
        034636  000720                                  BR      20$                     ;;SIMULATE CONTROL-U
                                                .DSABL  LSB

                                                ;;*************************************************************
                                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*CALL:
                                                ;*       RDCHR                          ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*       RETURN HERE                    ;;CHARACTER IS ON THE STACK
                                                ;*                                      ;;WITH PARITY BIT STRIPPED OFF
                                                ;

        034640  011646                          $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC
        034642  016666  000004  000002                  MOV     4(SP),2(SP)             ;;SAVE THE PS
        034650  105777  144304                  1$:     TSTB    @$TKS                   ;;WAIT FOR
        034654  100375                                  BPL     1$                      ;;A CHARACTER
        034656  117766  144300  000004                  MOVB    @$TKB,4(SP)             ;;READ THE TTY
        034664  042766  177600  000004                  BIC     #^C<177>,4(SP)          ;;GET RID OF JUNK IF ANY
        034672  026627  000004  000023                  CMP     4(SP),#23               ;;IS IT A CONTROL-S?
        034700  001013                                  BNE     3$                      ;;BRANCH IF NO
        034702  105777  144252                  2$:     TSTB    @$TKS                   ;;WAIT FOR A CHARACTER
        034706  100375                                  BPL     2$                      ;;LOOP UNTIL ITS THERE
        034710  117746  144246                          MOVB    @$TKB,-(SP)             ;;GET CHARACTER
        034714  042716  177600                          BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
        034720  022627  000021                          CMP     (SP)+,#21               ;;IS IT A CONTROL-Q?
        034724  001366                                  BNE     2$                      ;;IF NOT DISCARD IT
        034726  000750                                  BR      1$                      ;;YES, RESUME
        034730  026627  000004  000140          3$:     CMP     4(SP),#140              ;;IS IT UPPER CASE?
        034736  002407                                  BLT     4$                      ;;BRANCH IF YES
        034740  026627  000004  000175                  CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
        034746  003003                                  BGT     4$                      ;;BRANCH IF YES
        034750  042766  000040  000004                  BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
        034756  000002                          4$:     RTI                             ;;GO BACK TO USER
        034760     136     125     015          $CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL 'U'
        034765     136     107     015          $CNTLG: .ASCIZ  /^G/<15><12>            ;;CONTROL 'G'
        034772     015     012     123          $MSWR:  .ASCIZ  <15><12>/SWR = /
        035003     040     040     116          $MNEW:  .ASCIZ  /  NEW = /

                                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE

                                                ;;*************************************************************
                                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                                ;*WITH A RANGE OF 0 TO 2(+33)-1.
                                                ;*CALL:
                                                ;*       JSR     PC,$RAND               ;;CALL THE ROUTINE
                                                ;*       RETURN                         ;;RETURN HERE THE RANDOM
                                                ;*                                      ;;NUMBER WILL BE IN
                                                ;*                                      ;;$HINUM,$LONUM

        035014                                  $RAND:
```

```
    035014  010046                          MOV     R0,-(SP)         ;;PUSH RO ON STACK
    035016  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
    035020  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
    035022  013700  035114                  MOV     $LONUM,R0        ;;SET RO WITH LOW
    035026  013701  035112                  MOV     $HINUM,R1        ;;SET R1 WITH HIGH
    035032  012702  177771                  MOV     #-7,R2           ;;SET SHIFT COUNT
    035036  006300                  1$:     ASL     R0               ;;SHIFT RO LEFT AND
    035040  006101                          ROL     R1               ;;ROTATE CARRY INTO R1 AND
    035042  005202                          INC     R2               ;;CHECK FOR DONE
    035044  001374                          BNE     1$               ;;CONTINUE SHIFT LOOP
    035046  063700  035114                  ADD     $LONUM,R0        ;;ADD NUMBER TO MAKE X 129
    035052  005501                          ADC     R1               ;;PROPOGATE CARRY
    035054  063701  035112                  ADD     $HINUM,R1        ;;ADD NUMBER TO MAKE X 129
    035060  062700  001057                  ADD     #1057,R0         ;;ADD LOW CONSTANT
    035064  005501                          ADC     R1               ;;PROPOGATE CARRY
    035066  062701  047401                  ADD     #47401,R1        ;;ADD HIGH CONSTANT
    035072  010037  035114                  MOV     R0,$LONUM        ;;SAVE RO
    035076  010137  035112                  MOV     R1,$HINUM        ;;SAVE R1
    035102  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
    035104  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
    035106  012600                          MOV     (SP)+,R0         ;;POP STACK INTO RO
    035110  000207                          RTS     PC               ;;RETURN
    035112  176543                  $HINUM: .WORD   176543
    035114  123456                  $LONUM: .WORD   123456

                                    .SBTTL  SAVE AND RESTORE RO-R5 ROUTINES

                                    ;;*******************************************************************
                                    ;*SAVE RO-R5
                                    ;*CALL:
                                    ;*      SAVREG
                                    ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                    ;*
                                    ;*TOP---(+16)
                                    ;* +2---(+18)
                                    ;* +4---R5
                                    ;* +6---R4
                                    ;* +8---R3
                                    ;*+10---R2
                                    ;*+12---R1
                                    ;*+14---RO

    035116                          $SAVREG:
    035116  010046                          MOV     R0,-(SP)         ;;PUSH RO ON STACK
    035120  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
    035122  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
    035124  010346                          MOV     R3,-(SP)         ;;PUSH R3 ON STACK
    035126  010446                          MOV     R4,-(SP)         ;;PUSH R4 ON STACK
    035130  010546                          MOV     R5,-(SP)         ;;PUSH R5 ON STACK
    035132  016646  000022                  MOV     22(SP),-(SP)     ;;SAVE PS OF MAIN FLOW
    035136  016646  000022                  MOV     22(SP),-(SP)     ;;SAVE PC OF MAIN FLOW
    035142  016646  000022                  MOV     22(SP),-(SP)     ;;SAVE PS OF CALL
    035146  016646  000022                  MOV     22(SP),-(SP)     ;;SAVE PC OF CALL
    035152  000002                          RTI

                                    ;*RESTORE RO-R5
                                    ;*CALL:
```

```
                                  ;*      RESREG
035154                            $RESREG:
035154  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PC OF CALL
035160  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PS OF CALL
035164  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PC OF MAIN FLOW
035170  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PS OF MAIN FLOW
035174  012605                            MOV     (SP)+,R5          ;;POP STACK INTO R5
035176  012604                            MOV     (SP)+,R4          ;;POP STACK INTO R4
035200  012603                            MOV     (SP)+,R3          ;;POP STACK INTO R3
035202  012602                            MOV     (SP)+,R2          ;;POP STACK INTO R2
035204  012601                            MOV     (SP)+,R1          ;;POP STACK INTO R1
035206  012600                            MOV     (SP)+,R0          ;;POP STACK INTO R0
035210  000002                            RTI

                                  .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

035212  104412                    $DB2D:  SAVREG                      ;;SAVE REGISTERS
035214  016602  000002                    MOV     2(SP),R2            ;;PICKUP THE DATA POINTER
035220  012700  035372                    MOV     #$DECVL,R0          ;;GET ADDRESS OF "$DECVL" STRING
035224  010066  000002                    MOV     R0,2(SP)            ;;PUT ADDRESS OF ASCIZ STRING ON STACK
035230  012201                            MOV     (R2)+,R1            ;;PICKUP THE BINARY NUMBER
035232  012202                            MOV     (R2)+,R2
035234  012737  000012  035310            MOV     #10.,4$             ;;SET UP TO DO 10 CONVERSIONS
035242  012704  035322                    MOV     #$TNPWR,R4          ;;ADDRESS OF TEN POWER
035246  012705  035324                    MOV     #$TNPWR+2,R5
035252  005003                    1$:     CLR     R3                  ;;CLEAR PARTIAL
035254  161401                    2$:     SUB     (R4),R1             ;;SUBTRACT TEN POWER
035256  005602                            SBC     R2
035260  161502                            SUB     (R5),R2
035262  002402                            BLT     3$                  ;;BR IF TEN POWER TO LARGE
035264  005203                            INC     R3                  ;;ADD 1 TO PARTIAL
035266  000772                            BR      2$                  ;;LOOP
035270  062401                    3$:     ADD     (R4)+,R1            ;;RESTORE SUBTRACTED VALUE
035272  005502                            ADC     R2
035274  062402                            ADD     (R4)+,R2
035276  022525                            CMP     (R5)+,(R5)+         ;;MOVE TO NEXT TEN POWER
035300  052703  000060                    BIS     #'0,R3              ;;CHANGE PARTIAL TO ASCII
035304  110320                            MOVB    R3,(R0)+            ;;SAVE IT
035306  005327                            DEC     (PC)+               ;;DONE?
035310  000000                    4$:     .WORD   0
035312  001357                            BNE     1$                  ;;BR IF NO
035314  105020                            CLRB    (R0)+               ;;TERMINATOR
035316  104413                            RESREG                      ;;RESTORE REGISTERS
035320  000207                            RTS     PC                  ;;RETURN
035322  145000                    $TNPWR: 145000                      ;;1.0E09
035324  035632                            35632
```

```
        035326  160400                         160400                  ;;1.0E08
        035330  002765                         2765
        035332  113200                         113200                  ;;1.0E07
        035334  000230                         230
        035336  041100                         041100                  ;;1.0E06
        035340  000017                         17
        035342  103240                         103240                  ;;1.0E05
        035344  000001                         1
        035346  023420                         23420                   ;;1.0E04
        035350  000000                         0
        035352  001750                         1750                    ;;1.0E03
        035354  000000                         0
        035356  000144                         144                     ;;1.0E02
        035360  000000                         0
        035362  000012                         12                      ;;1.0E01
        035364  000000                         0
        035366  000001                         1                       ;;1.0E00
        035370  000000                         0
        035372                         $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING

                                       .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

                                       ;;*************************************************************
                                       ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
                                       ;*UNSIGNED OCTAL ASCIZ NUMBER.
                                       ;*CALL
                                       ;*       MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                       ;*       JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
                                       ;*       RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

        035406  104412                 $DB2O:  SAVREG                  ;;SAVE ALL REGISTERS
        035410  016601  000002                 MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
        035414  012705  035525                 MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
        035420  012704  000014                 MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
        035424  012703  177770                 MOV     #^C7,R3         ;;MASK
        035430  012100                         MOV     (R1)+,R0        ;;LOWER WORD
        035432  012101                         MOV     (R1)+,R1        ;;HIGH WORD
        035434  005002                         CLR     R2              ;;TERMINATOR
        035436  110245                 1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
        035440  010002                         MOV     R0,R2           ;;GET THIS DIGIT
        035442  005304                         DEC     R4              ;;COUNT THIS CHARACTER
        035444  003007                         BGT     3$              ;;BR IF NOT THE LAST DIGIT
        035446  001405                         BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
        035450  005205                         INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
        035452  010566  000002                 MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
        035456  104413                         RESREG                  ;;RESTORE ALL REGISTERS
        035460  000207                         RTS     PC              ;;RETURN TO USER
        035462  006203                 2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
        035464  006001                 3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
        035466  006000                         ROR     R0              ;;      THE NEXT OCTAL DIGIT
        035470  006001                         ROR     R1
        035472  006000                         ROR     R0
        035474  006001                         ROR     R1
        035476  006000                         ROR     R0
        035500  040302                         BIC     R3,R2           ;;MASK OUT ALL JUNK
        035502  062702  000060                 ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
```

```
     035506  000753                          BR      1$                  ;;GO PUT IT IN THE DATA TABLE
     035510                         $OCTVL: .BLKB   14.                 ;;RESERVE DATA TABLE

                                    .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.

     035526  010046                 $TRAP:  MOV     R0,-(SP)            ;;SAVE R0
     035530  016600  000002                 MOV     2(SP),R0            ;;GET TRAP ADDRESS
     035534  005740                         TST     -(R0)               ;;BACKUP BY 2
     035536  111000                         MOVB    (R0),R0             ;;GET RIGHT BYTE OF TRAP
     035540  006300                         ASL     R0                  ;;POSITION FOR INDEXING
     035542  016000  035562                 MOV     $TRPAD(R0),R0       ;;INDEX TO TABLE
     035546  000200                         RTS     R0                  ;;GO TO ROUTINE

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

     035550  011646                 $TRAP2: MOV     (SP),-(SP)          ;;MOVE THE PC DOWN
     035552  016666  000004  000002         MOV     4(SP),2(SP)         ;;MOVE THE PSW DOWN
     035560  000002                         RTI                         ;;RESTORE THE PSW

                                    .SBTTL  TRAP TABLE

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

                                    ;       ROUTINE
                                    ;       -------
     035562  035550                 $TRPAD: .WORD   $TRAP2
     035564  032606                         $TYPE   ;;CALL=TYPE         TRAP+1(104401)  TTY TYPEOUT ROUTINE
     035566  033730                         $TYPOC  ;;CALL=TYPOC        TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
     035570  033704                         $TYPOS  ;;CALL=TYPOS        TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
     035572  033744                         $TYPON  ;;CALL=TYPON        TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
     035574  034132                         $TYPDS  ;;CALL=TYPDS        TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

     035576  034426                         $GTSWR  ;;CALL=GTSWR        TRAP+6(104406)  GET SOFT-SWR SETTING

     035600  034356                         $CKSWR  ;;CALL=CKSWR        TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
     035602  034640                         $RDCHR  ;;CALL=RDCHR        TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
     035604  031766                         $RDLIN  ;;CALL=RDLIN        TRAP+11(104411) TTY TYPEIN STRING ROUTINE
     035606  035116                         $SAVREG ;;CALL=SAVREG       TRAP+12(104412) SAVE R0-R5 ROUTINE
     035610  035154                         $RESREG ;;CALL=RESREG       TRAP+13(104413) RESTORE R0-R5 ROUTINE
     035612  031212                         $DSPLY  ;;CALL=DISPLY       TRAP+14(104414) ROUTINE TO TYPE ERROR MESSAGES
             000032                 $TERM=.-$TRPAD
```

```
                                        .SBTTL  SINGLE/DUAL PORT RH/RM DRIVER (REV 6.2) FEBRUARY 1980

                                        ;NEW DRIVE TYPE ID FOR RM02 **********
                                        ;10-AUG-77 ************
                                        ;10-MAR-78 THE SC, SC5 CHANGES
                                        ;NEW DRIVE TYPE ID FOR RM05 **********
                                        ;FEBRUARY 1980 ***********

                                        ;COPYRIGHT (C) 1977
                                        ;DIGITAL EQUIPMENT CORP.
                                        ;MAYNARD, MA 01754
                                        ;AUTHOR(S): JIM LACEY/CHUCK HESS/

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

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

035614  000000  000000  000000  RMERRS: .WORD   0,0,0,0

                                        ;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

035624     000                          DRVACT: .BYTE   0               ;DRIVE 0
035625     000                                  .BYTE   0               ;DRIVE 1
035626     000                                  .BYTE   0               ;DRIVE 2
035627     000                                  .BYTE   0               ;DRIVE 3
035630     000                                  .BYTE   0               ;DRIVE 4
035631     000                                  .BYTE   0               ;DRIVE 5
035632     000                                  .BYTE   0               ;DRIVE 6
035633     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

035634     000                          DRVSTA: .BYTE   0               ;DRIVE 0
035635     000                                  .BYTE   0               ;DRIVE 1
035636     000                                  .BYTE   0               ;DRIVE 2
035637     000                                  .BYTE   0               ;DRIVE 3
035640     000                                  .BYTE   0               ;DRIVE 4
035641     000                                  .BYTE   0               ;DRIVE 5
035642     000                                  .BYTE   0               ;DRIVE 6
035643     000                                  .BYTE   0               ;DRIVE 7

                                        ;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
                                                ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
                                                ;DRVTYP=7 IF DRIVE IS RM05 ******
                                                ;DRVTYP=5 IF DRIVE IS RM02 ******
                                                ;DRVTYP=4 IF DRIVE IS RM03
                                                ;DRVTYP=-1 IF NOT RM05/3/2
```

```
 035644       000                  DRVTYP: .BYTE    0                 ;DRIVE 0
 035645       000                          .BYTE    0                 ;DRIVE 1
 035646       000                          .BYTE    0                 ;DRIVE 2
 035647       000                          .BYTE    0                 ;DRIVE 3
 035650       000                          .BYTE    0                 ;DRIVE 4
 035651       000                          .BYTE    0                 ;DRIVE 5
 035652       000                          .BYTE    0                 ;DRIVE 6
 035653       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

 035654       000                  DPINT:  .BYTE    0                 ;DRIVE 0
 035655       000                          .BYTE    0                 ;DRIVE 1
 035656       000                          .BYTE    0                 ;DRIVE 2
 035657       000                          .BYTE    0                 ;DRIVE 3
 035660       000                          .BYTE    0                 ;DRIVE 4
 035661       000                          .BYTE    0                 ;DRIVE 5
 035662       000                          .BYTE    0                 ;DRIVE 6
 035663       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

 035664       000                  DPRQS:  .BYTE    0                 ;DRIVE 0
 035665       000                          .BYTE    0                 ;DRIVE 1
 035666       000                          .BYTE    0                 ;DRIVE 2
 035667       000                          .BYTE    0                 ;DRIVE 3
 035670       000                          .BYTE    0                 ;DRIVE 4
 035671       000                          .BYTE    0                 ;DRIVE 5
 035672       000                          .BYTE    0                 ;DRIVE 6
 035673       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.

 035674    000000                  TRNSWT: .WORD    0

                                   ;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.

 035676    000000                  SRCHWT: .WORD    0

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

 035700       000                  ACTDRV: .BYTE    0

                                   ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
```

```
                                                ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
                                                ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE

 035701      000                ACTSTR: .BYTE   0

                                ;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

 035702      000                ULDFLG: .BYTE   0               ;DRIVE 0
 035703      000                        .BYTE   0               ;DRIVE 1
 035704      000                        .BYTE   0               ;DRIVE 2
 035705      000                        .BYTE   0               ;DRIVE 3
 035706      000                        .BYTE   0               ;DRIVE 4
 035707      000                        .BYTE   0               ;DRIVE 5
 035710      000                        .BYTE   0               ;DRIVE 6
 035711      000                        .BYTE   0               ;DRIVE 7

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

 035712      000                LACNT:  .BYTE   0               ;DRIVE 0
 035713      000                        .BYTE   0               ;DRIVE 1
 035714      000                        .BYTE   0               ;DRIVE 2
 035715      000                        .BYTE   0               ;DRIVE 3
 035716      000                        .BYTE   0               ;DRIVE 4
 035717      000                        .BYTE   0               ;DRIVE 5
 035720      000                        .BYTE   0               ;DRIVE 6
 035721      000                        .BYTE   0               ;DRIVE 7

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

 035722   000000                SAVEFG: .WORD   0

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

 035724   177777                SEEKFG: .WORD   -1

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

 035726   177777                TIMER:  .WORD   -1              ;DRIVE 0
 035730   177777                        .WORD   -1              ;DRIVE 1
 035732   177777                        .WORD   -1              ;DRIVE 2
 035734   177777                        .WORD   -1              ;DRIVE 3
 035736   177777                        .WORD   -1              ;DRIVE 4
 035740   177777                        .WORD   -1              ;DRIVE 5
 035742   177777                        .WORD   -1              ;DRIVE 6
```

```
035744  177777                          .WORD   -1                      ;DRIVE 7

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

035746  177777                  DTUW:   .WORD   -1

                                ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
                                        ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
                                        ;ATTENTION BIT

035750     001                  ATABIT: .BYTE   1       ;DRIVE 0
035751     002                          .BYTE   2       ;DRIVE 1
035752     004                          .BYTE   4       ;DRIVE 2
035753     010                          .BYTE   10      ;DRIVE 3
035754     020                          .BYTE   20      ;DRIVE 4
035755     040                          .BYTE   40      ;DRIVE 5
035756     100                          .BYTE   100     ;DRIVE 6
035757     200                          .BYTE   200     ;DRIVE 7

                                ;FSRM TO RH11/RH70 'MASSBUS CONTROL BUS PARITY ERRORS'' (MCPE) ALLOWED BEFORE
                                ;CALLING IT FATAL (MCPEMX=1 WORD)

035760  000003                  MCPEMX: .WORD   3

                                ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH/RM),
                                ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).

035762  176700                  RMADR:  .WORD   176700
035764  000254  000240          RMVEC:  .WORD   254,5*32.

                                ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
035770  000004                  MXLACT: .WORD   4

                                ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
035772  001000                  MXDLTA: .WORD   8.*64.

                                ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
035774  000200                  MNDLTA: .WORD   2*64.

                                ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
035776  000005                  MXWNDW: .WORD   5

                                ;DEFINITIONS OF THE RH/RM ADDRESS INDEXES

        000000                  RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
        000002                  RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
        000004                  RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
        000006                  RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
        000010                  RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
        000012                  RMDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
        000014                  RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
        000016                  RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
        000020                  RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
        000022                  RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
        000024                  RMMR1=24                ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
        000026                            RMDT=26                        ;DRIVE TYPE REGISTER (DRIVE REG. 06)
        000030                            RMSN=30                        ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
        000032                            RMOF=32                        ;OFFSET REGISTER (DRIVE REG. 11)
        000034                            RMDC=34                        ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
        000036                            RMHR=36                        ;DUMMY  ADDRESS REGISTER (DRIVE REG. 13)
        000040                            RMMR2=40                       ;MAINTENANCE REGISTER #2
        000042                            RMER2=42                       ;ERROR REGISTER #2 (DRIVE REG. 15)
        000044                            RMEC1=44                       ;ECC POSITION REGISTER (DRIVE REG. 16)
        000046                            RMEC2=46                       ;ECC PATTERN REGISTER (DRIVE REG.  17)

                                          .SBTTL  RH/RM DRIVER INITIALIZATION CODE

                                          ;THIS ROUTINE WILL DETERMINE WHICH RM DRIVES ARE
                                          ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                          ;TO THE PROPER STATE FOR EACH DRIVE.
                                          ;NOTE: THIS ROUTINE CALLS DRVINT
                                          ;
                                          ;CALL
                                          ;
                                          ;       JSR     PC,RMINIT
                                          ;       RETURN
                                          ;
                                          ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                          ;
036000  104412                            RMINIT: SAVREG                         ;SAVE R0 - R5
036002  013746  177776                            MOV     PS,-(SP)               ;SAVE THE PRESENT PROCESSOR STATUS
036006  012737  000240  177776                    MOV     #<5*32.>,PS            ;CHANGE THE PRIORITY TO 5
036014  004737  043746                            JSR     PC,CLRQUE              ;CLEAR ALL REQUEST QUEUES
036020  012701  035614                            MOV     #RMERRS,R1             ;FIRST ADDRESS TO BE CLEARED
036024  012702  035724                            MOV     #SEEKFG,R2             ;LAST ADDRESS TO BE CLEARED
036030  005021                            1$:     CLR     (R1)+                  ;CLEAR
036032  020102                                    CMP     R1,R2                  ;ARE WE DONE?
036034  103775                                    BLO     1$                     ;BR IF NO
036036  012702  035746                            MOV     #DTUW,R2               ;LAST ADDRESS
036042  012721  177777                    2$:     MOV     #-1,(R1)+              ;INITIALIZE
036046  020102                                    CMP     R1,R2                  ;DONE?
036050  101774                                    BLOS    2$                     ;LOOP IF NO
036052  005037  035634                            CLR     DRVSTA                 ;SET ALL DRIVES TO OFFLINE
036056  005037  035636                            CLR     DRVSTA+2
036062  005037  035640                            CLR     DRVSTA+4
036066  005037  035642                            CLR     DRVSTA+6
036072  013703  035764                            MOV     RMVEC,R3               ;SETUP THE RH/RM VECTOR
036076  012723  040634                            MOV     #ISR,(R3)+
036102  013713  035766                            MOV     RMVEC+2,(R3)
036106  013704  035762                            MOV     RMADR,R4               ;FIRST ADDRESS OF RH/RM
036112  012764  000040  000010                    MOV     #CLR,RMCS2(R4)         ;MASSBUS INIT
036120  005001                                    CLR     R1                     ;START WITH DRIVE 0
036122  004037  036212                    3$:     JSR     R0,DRVINT              ;INIT THE DRIVE
036126  000401                                    BR      4$                     ;'DVA' NOT SET OR PARITY ERROR
036130  000402                                    BR      5$                     ;NORMAL RETURN
036132  105061  035634                    4$:     CLRB    DRVSTA(R1)             ;SET DRIVE STATUS TO OFFLINE
036136  005201                            5$:     INC     R1                     ;GO TO NEXT DRIVE
036140  042701  177770                            BIC     #^C7,R1                ;MASK OUT UNUSED BITS
036144  001366                                    BNE     3$                     ;BR IF MORE DRIVES TO GO
036146  012701  000007                            MOV     #7,R1                  ;START WITH DRIVE 7
036152  005037  177776                            CLR     PS                     ;CLEAR THE PROCESSOR STATUS
036156  105761  035654                    6$:     TSTB    DPINT(R1)              ;WAITING FOR DRIVE TO SWITCH PORTS ?
```

```
036162  001405                           BEQ     8$                 ;BR NOT WAITING
036164  004737  043402                   JSR     PC,SET.IE          ;SET INTERRUPT
036170  105761  035654          7$:      TSTB    DPINT(R1)          ;DRIVE SWITCHED PORTS ?
036174  001375                           BNE     7$                 ;BR IF NOT
036176  005301                  8$:      DEC     R1                 ;GO TO THE NEXT DRIVE
036200  100366                           BPL     6$                 ;CHECK NEXT DRIVE
036202  012637  177776                   MOV     (SP)+,PS           ;RESTORE THE PROCESSOR STATUS
036206  104413                           RESREG                     ;RESTORE R0 - R5
036210  000207                           RTS     PC                 ;BYE-BYE

                                ;DRIVE INITILIZATION ROUTINE
                                ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
                                ;AN RM05/3/2. IF IT IS, A ''READ-IN PRESET'' IS ISSUED AND FMT16
                                ;IS SET TO A ''1''. THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
                                ;INSURE THEY ARE ALL ON A ''1''. AND DEPENDING ON THEIR STATE,
                                ;DRVSTA IS SET TO THE PROPER CONDITION.
                                ;CALL
                                ;        MOV     #DRVNUM,R1         ;DRIVE NUMBER TO R1
                                ;        MOV     RMADR,R4           ;UNIBUS ADDRESS OF RH/RM (RMCS1)
                                ;        JSR     R0,DRVINT          ;CALLED BY A JSR
                                ;        RETURN1                    ;ERROR OCCURRED (PARITY)
                                ;        RETURN2                    ;NORMAL RETURN
                                ;
                                ;
036212  010546                  DRVINT:  MOV     R5,-(SP)           ;SAVE R5
036214  105061  035634          DULP:    CLRB    DRVSTA(R1)         ;START DRIVE STATUS AS OFFLINE
036220  105061  035644                   CLRB    DRVTYP(R1)         ;CLEAR THE DRIVE TYPE INDICATOR
036224  105061  035702                   CLRB    ULDFLG(R1)         ;CLEAR THE UNLOAD FLAG
036230  010164  000010                   MOV     R1,RMCS2(R4)       ;SELECT A DRIVE
036234  112764  000111  000000           MOVB    #111,RMCS1(R4)     ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
036242  032764  010000  000010           BIT     #BIT12,RMCS2(R4)           ;NONEXISTENT DRIVE?
036250  001403                           BEQ     1$                 ;NO
036252  004737  043402                   JSR     PC,SET.IE          ;GO SET ''IE'' WITHOUT A ''TRE''
036256  000520                           BR      6$                 ;LEAVE THIS ROUTINE

036260  105061  035634          1$:      CLRB    DRVSTA(R1)         ;SET DRIVE STATUS TO OFFLINE
036264  032764  004000  000000           BIT     #BIT11,RMCS1(R4)           ;SEE IF DRIVE AVAILABLE
036272  001750                           BEQ     DULP               ;BR IF DRIVE NOT AVAILABLE
036274  004037  042722                   JSR     R0,RD.RM           ;READ THE DRIVE TYPE REG.
036300  000026                           RMDT
036302  036544                           8$                         ;ERROR RETURN ADDRESS
036304  012605                           MOV     (SP)+,R5           ;PUT DRIVE TYPE IN R5
036306  112761  000004  035644           MOVB    #4,DRVTYP(R1)      ;SET RM03 INDICATOR
036314  022705  020024                   CMP     #20024,R5          ;SINGLE PORT RM03 ?
036320  001431                           BEQ     2$                 ;BR IF YES
036322  022705  024024                   CMP     #24024,R5          ;DUAL PORT RM03 ?
036326  001426                           BEQ     2$                 ;BR IF YES
036330  112761  000005  035644           MOVB    #5,DRVTYP(R1)      ;SET RM02 INDICATOR
036336  022705  020025                   CMP     #20025,R5          ;SINGLE PORT RM02 ?
036342  001420                           BEQ     2$                 ;BR IF SO
036344  022705  024025                   CMP     #24025,R5          ;DUAL PORT RM02 ?
036350  001415                           BEQ     2$                 ;BR IF SO
036352  112761  000007  035644           MOVB    #7,DRVTYP(R1)      ;SET RM05 INDICATOR
036360  022705  020027                   CMP     #20027,R5          ;SINGLE PORT RM05 ?
036364  001407                           BEQ     2$                 ;BR IF YES
036366  022705  024027                   CMP     #24027,R5          ;DUAL PORT RM05 ?
036372  001404                           BEQ     2$                 ;BR IF YES
```

```
036374  112761  177777  035644          MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
036402  000446                          BR      6$              ;EXIT

036404  012746  000121          2$:     MOV     #121,-(SP)      ;DO A ''READ-IN PRESET''
036410  004037  043076                  JSR     R0,WRT.RM
036414  000000                          RMCS1
036416  036544                          8$
036420  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT16=1
036424  004037  043076                  JSR     R0,WRT.RM
036430  000032                          RMOF
036432  036544                          8$
036434  004037  042722                  JSR     R0,RD.RM        ;READ RMDS
036440  000012                          RMDS
036442  036544                          8$
036444  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
036446  100015                          BPL     4$              ;BR IF ATA=0
036450  116164  035750  000016          MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
036456  004037  042722                  JSR     R0,RD.RM        ;FIND OUT WHY ATA=1
036462  000014                          RMER1
036464  036544                          8$
036466  006126                          ROL     (SP)+           ;IS IT UNSAFE?
036470  100004                          BPL     4$              ;BR IF NOT
036472  112761  177777  035634          MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
036500  000407                          BR      6$              ;EXIT

036502  005105                  4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
036504  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
036510  001003                          BNE     6$              ;BR IF MOL, DPR, DRY, OR VV IS CLEAR
036512  112761  000001  035634          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
036520  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
036522  000410                          BR      8$              ;EXIT
036524  006301                  7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
036526  012761  060000  035726          MOV     #60000,TIMER(R1)        ;START 2 SEC TIMER
036534  006201                          ASR     R1              ;RESTORE R1
036536  112761  177777  035654          MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INIDICATOR
036544  012605                  8$:     MOV     (SP)+,R5        ;RESTORE R5
036546  000200                          RTS     R0              ;EXIT

                                ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
                                ;
                                ;CALL
                                ;
                                ;       JSR     R0,RM05         ;CALL THE RM05 DRIVER
                                ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
                                ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
                                ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
                                                                ;IS AN ERROR CONDITION

036550  013746  177776          RM05:   MOV     PS,-(SP)        ;SAVE THE CALLING STATUS
036554  013737  035766  177776          MOV     RMVEC+2,PS      ;DON'T ALLOW ANY RM INTERRUPTS
036562  112737  000001  035700          MOVB    #1,ACTDRV       ;SET ''ACTIVE DRIVER'' FLAG
036570  104412                          SAVREG                  ;SAVE R0 - R5
036572  011002                          MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
036574  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
036600  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
036602  013704  035762                  MOV     RMADR,R4        ;UNIBUS ADDRESS OF RMCS1
036606  105761  035634                  TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
```

```
036612  003014                          BGT     1$                      ;BR IF ONLINE
036614  105761  035702                  TSTB    ULDFLG(R1)              ;UNLOAD COMMAND IN QUEUE?
036620  001036                          BNE     3$                      ;BR IF YES
036622  105761  035654                  TSTB    DPINT(R1)               ;TRYING TO INIT THE DRIVE
036626  001042                          BNE     5$                      ;BR IF YES
036630  004037  036212                  JSR     R0,DRVINT               ;GO INIT. THE DRIVE
036634  000434                          BR      4$                      ;ERROR RETURN
036636  105761  035634                  TSTB    DRVSTA(R1)              ;IS DRIVE STATUS ONLINE?
036642  003445                          BLE     6$                      ;BR IF NOT
036644  105761  035664          1$:     TSTB    DPRQS(R1)               ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
036650  001031                          BNE     5$                      ;BR IF YES
036652  010164  000010                  MOV     R1,RMCS2(R4)            ;SELECT THE DRIVE
036656  004037  044044                  JSR     R0,DRVQUE               ;PUT THIS REQUEST IN QUEUE
036662  000460                          BR      9$                      ;QUEUE IS FULL
036664  122762  000103  000002          CMPB    #103,2(R2)              ;IS THIS REQ. FOR AN UNLOAD?
036672  001003                          BNE     2$                      ;BR IF NO
036674  112761  177777  035702          MOVB    #-1,ULDFLG(R1)          ;SET THE ''UNLOAD IN QUEUE'' FLAG
036702  105761  035624          2$:     TSTB    DRVACT(R1)              ;IS THIS DRIVE ACTIVE?
036706  001043                          BNE     8$                      ;BR IF YES
036710  004737  037042                  JSR     PC,OPT                  ;CALL THE OPTIMIZER
036714  000440                          BR      8$
036716  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2)             ;SET THE ''UNLOAD IN QUEUE'' ERROR FLAG
036724  000434                          BR      8$                      ;EXIT
036726  004737  040122          4$:     JSR     PC,CI7                  ;GO HANDLE THE PARITY ERROR
036732  000431                          BR      8$
036734  004037  044044          5$:     JSR     R0,DRVQUE               ;PUT REQUEST IN QUEUE
036740  000431                          BR      9$                      ;QUEUE IS FULL
036742  032714  000100                  BIT     #BIT06,(R4)             ;IE BIT SET ?
036746  001023                          BNE     8$                      ;YES
036750  004737  043402                  JSR     PC,SET.IE               ;SET THE INTERRUPT
036754  000420                          BR      8$                      ;RETURN
036756  105761  035634          6$:     TSTB    DRVSTA(R1)              ;SEE IF DRIVE OFFLINE OR UNSAFE
036762  002412                          BLT     7$                      ;BR IF UNSAFE
036764  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)             ;SET OFFLINE ERROR INDICATOR
036772  105761  035644                  TSTB    DRVTYP(R1)              ;SEE IF OFFLINE OR NONEXISTENT
036776  001007                          BNE     8$                      ;BR IF OFFLINE
037000  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)             ;REPORT DRIVE NONEXISTENT
037006  000403                          BR      8$                      ;GO TO EXIT
037010  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2)             ;DRIVE IS UNSAFE
037016  104413                  8$:     RESREG                          ;RESTORE R0 - R5
037020  005720                          TST     (R0)+                   ;SETUP FOR NORMAL RETURN
037022  000401                          BR      10$                     ;FINISH UP, THEN EXIT
037024  104413                  9$:     RESREG                          ;RESTORE R0 - R5
037026  005720                  10$:    TST     (R0)+                   ;CORRECT THE RETURN ADDRESS
037030  105037  035700                  CLRB    ACTDRV                  ;CLEAR ''ACTIVE DRIVER'' FLAG
037034  012637  177776                  MOV     (SP)+,PS                ;RETURN ''PS'' TO USER LEVEL
037040  000200                          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
                                ;
037042  104412                  OPT:    SAVREG                          ;SAVE R0 - R5
037044  013746  177776                  MOV     PS,-(SP)                ;SAVE PROC. STATUS
037050  146137  035750  035676          BICB    ATABIT(R1),SRCHWT               ;CLEAR LA SEACH FLAG
```

```
037056  105061  035664                    CLRB    DPRQS(R1)        ;RESET THE PORT REQ FLAG ****
037062  004737  044120                    JSR     PC,GETREQ        ;GET ''DPB'' POINTER OF REQUEST
037066  005702                            TST     R2               ;IS THERE A REQUEST IN QUEUE?
037070  001472                            BEQ     7$               ;NO--BR TO EXIT
037072  010164  000010                    MOV     R1,RMCS2(R4)     ;LOAD THE DRIVE ADDRESS *******
037076  012764  000111  000000            MOV     #111,RMCS1(R4)   ;CLEAR THE DRIVE
037104  032764  004000  000000            BIT     #BIT11,RMCS1(R4)         ;DVA SET ?
037112  001446                            BEQ     5$               ;TO PROT REQUEST ,IF NOT
037114  105761  035634          10$:      TSTB    DRVSTA(R1)       ;IS DRIVE ONLINE?
037120  003014                            BGT     1$               ;YES
037122  004737  044142                    JSR     PC,POPQUE        ;NO--REMOVE REQUEST FROM QUEUE
037126  012762  140000  000016            MOV     #BIT15!BIT14,16(R2)      ;SET OFFLINE STATUS/ERROR INDICATOR
037134  105761  035634                    TSTB    DRVSTA(R1)       ;IS DRIVE UNSAFE ?
037140  100053                            BPL     8$               ;BR TO EXIT IF NOT
037142  012762  110000  000016            MOV     #BIT15!BIT12,16(R2)      ;SET UNSAFE STATUS/ERROR INDICATOR
037150  000447                            BR      8$               ;BR TO EXIT
037152                          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
037152  122762  000150  000002            CMPB    #150,2(R2)       ;IS THE REQUEST FOR I/O?
037160  002403                            BLT     2$               ;YES
037162  004737  037506                    JSR     PC,CI4           ;CALL THE COMMAND INITIATOR
037166  000440                            BR      8$               ;BR TO EXIT
037170  005737  035746          2$:       TST     DTUW             ;DATA TRANSFER UNDERWAY?
037174  002012                            BGE     4$               ;YES--GO START A SEARCH
037176  005737  035724                    TST     SEEKFG           ;DO IMPLIED SEEKS?
037202  100404                            BMI     3$               ;YES
037204  004037  040460                    JSR     R0,LA            ;NO--DO LOOK AHEAD
037210  000427                            BR      8$               ;RETURN HERE ON A PARITY ERROR
037212  000403                            BR      4$               ;GO START A SEARCH
037214  004737  037300          3$:       JSR     PC,CI1           ;START A DATA TRANSFER
037220  000423                            BR      8$
037222  004737  037406          4$:       JSR     PC,CI3           ;START A SEARCH
037226  000420                            BR      8$               ;GO TO THE EXIT
037230  112761  177777  035664  5$:       MOVB    #-1,DPRQS(R1)    ;SET PORT REQUEST INDICATOR
037236  010103                            MOV     R1,R3            ;SET UP TO ADDRESS WORDS
037240  006303                            ASL     R3               ;CONVERT TO WORD INDEX
037242  012763  060000  035726            MOV     #60000,TIMER(R3)         ;START 10 SEC TIMER
037250  000402                            BR      7$               ;EXIT
037252  004737  040122          6$:       JSR     PC,CI7           ;PROCESS THE PARITY ERROR
037256  032714  000100          7$:       BIT     #BIT06,(R4)      ;SEE IF 'IE' ALREADY SET
037262  001002                            BNE     8$               ;BR IF SET
037264  004737  043402                    JSR     PC,SET.IE        ;SET ''IE'' WITHOUT A ''TRE''
037270  012637  177776          8$:       MOV     (SP)+,PS         ;RESTORE PROC. STATUS
037274  104413                            RESREG                   ;RESTORE R0 - R5
037276  000207                            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

037300  004737  044142          CI1:    JSR     PC,POPQUE         ;REMOVE REQUEST FROM ''DRIVES WAIT'' QUEUE
037304  010237  035674                  MOV     R2,TRNSWT         ;PUT REQ. IN TRANSFER WAIT QUEUE
037310  010203                          MOV     R2,R3             ;DPB ADDRESS TO R3
037312  013704  035762                  MOV     RMADR,R4          ;RMCS1 ADDRESS
037316  010164  000010                  MOV     R1,RMCS2(R4)      ;SELECT DRIVE
037322  062703  000004                  ADD     #4,R3             ;DESIRED WORD COUNT
037326  062704  000002                  ADD     #2,R4             ;RMWC ADDRESS
037332  012324                          MOV     (R3)+,(R4)+       ;LOAD WORD COUNT
037334  012324                          MOV     (R3)+,(R4)+       ;LOAD BUFFER ADDRESS
037336  012346                          MOV     (R3)+,-(SP)       ;LOAD SECTOR AND TRACK
037340  004037  043076                  JSR     R0,WRT.RM         ;CALL THE LOAD(WRITE) ROUTINE
037344  000006                          RMDA                      ;INDEX OF REGISTER TO LOAD
037346  040122                          CI7                       ;ERROR RETURN ADDRESS
037350  012346                          MOV     (R3)+,-(SP)       ;LOAD CYLINDER ADDRESS
037352  004037  043076                  JSR     R0,WRT.RM
037356  000034                          RMDC
037360  040122                          CI7
037362  016246  000002                  MOV     2(R2),-(SP)       ;LOAD ''COMMAND+GO'', ''A17&A16'', AND ''PSEL''
037366  004037  043076                  JSR     R0,WRT.RM
037372  000000                          RMCS1
037374  040122                          CI7
037376  010137  035746                  MOV     R1,DTUW           ;SET ''DATA TRANSFER UNDERWAY''
037402  000137  040064                  JMP     CI5
037406  013704  035762          CI3:    MOV     RMADR,R4          ;RMCS1 ADDRESS
037412  010164  000010                  MOV     R1,RMCS2(R4)      ;SELECT DRIVE
037416  016246  000012                  MOV     12(R2),-(SP)      ;DESIRED CYLINDER ADDRESS
037422  004037  043076                  JSR     R0,WRT.RM
037426  000034                          RMDC
037430  040122                          CI7
037432  116203  000010                  MOVB    10(R2),R3         ;PICKUP SECTOR ADDRESS
                                ;       SUB     MXWNDW,R3         ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
                                ;       BGE     1$
                                ;       ADD     #32.,R3
037436  010346                  1$:     MOV     R3,-(SP)          ;COMBINE THE ADJUSTED SECTOR WITH
037440  042716  177740                  BIC     #177740,(SP)      ;CHOP OF ALL EXENTED BITS ,IF ANY
037444  116266  000011  000001          MOVB    11(R2),1(SP)      ;THE DESIRED TRACK
037452  004037  043076                  JSR     R0,WRT.RM         ;LOAD DESIRED TRACK & SECTOR
037456  000006                          RMDA
037460  040122                          CI7
037462  012746  000131                  MOV     #131,-(SP)        ;START A SEARCH
037466  004037  043076                  JSR     R0,WRT.RM
037472  000000                          RMCS1
037474  040122                          CI7
037476  156137  035750  035676          BISB    ATABIT(R1),SRCHWT         ;SET ''SEARCH WAIT'' KEY
037504  000567                          BR      CI5
037506  013704  035762          CI4:    MOV     RMADR,R4          ;RMCS1 ADDRESS
037512  010164  000010                  MOV     R1,RMCS2(R4)      ;SELECT DRIVE
037516  116203  000002                  MOVB    2(R2),R3          ;PICKUP THE REQUESTED COMMAND
037522  122703  000131                  CMPB    #131,R3           ;IS IT A SEARCH COMMAND?
037526  001007                          BNE     1$                ;BR IF NO
037530  016246  000010                  MOV     10(R2),-(SP)      ;LOAD DESIRED TRACK & SECTOR
037534  004037  043076                  JSR     R0,WRT.RM
```

```
037540  000006                                  RMDA
037542  040122                                  CI7
037544  000403                                  BR      2$                      ;GO LOAD CYLINDER
037546  122703  000105                  1$:     CMPB    #105,R3                 ;IS IT A SEEK COMMAND
037552  001007                                  BNE     3$                      ;BR IF NO
037554  016246  000012                  2$:     MOV     12(R2),-(SP)            ;LOAD DESIRED CYLINDER
037560  004037  043076                          JSR     R0,WRT.RM
037564  000034                                  RMDC
037566  040122                                  CI7
037570  000546                                  BR      CI6
037572  122703  000115                  3$:     CMPB    #115,R3                 ;IS IT AN ''OFFSET'' COMMAND?
037576  001013                                  BNE     4$                      ;BR IF NO
037600  004037  042722                          JSR     R0,RD.RM                ;MERGE THE OFFSET VALUE INTO RMOF
037604  000032                                  RMOF                            ;BUT DON'T CHANGE THE UPPER
037606  040122                                  CI7
037610  116216  000001                          MOVB    1(R2),(SP)              ;BYTE WHEN LOADING THE
037614  004037  043076                          JSR     R0,WRT.RM               ;REGISTER (RMOF)
037620  000032                                  RMOF
037622  040122                                  CI7
037624  000530                                  BR      CI6                     ;GO START THE COMMAND
037626  122703  000107                  4$:     CMPB    #107,R3                 ;IS IT A ''RECALIBRATE'' COMMAND?
037632  001525                                  BEQ     CI6                     ;BR IF YES
037634  122703  000117                          CMPB    #117,R3                 ;IS IT A RETURN TO CENTER?
037640  001522                                  BEQ     CI6                     ;BR IF YES
037642  122703  000103                          CMPB    #103,R3                 ;IS IT AN ''UNLOAD'' COMMAND?
037646  001016                                  BNE     5$                      ;BR IF NO
037650  112761  000001  035624                  MOVB    #1,DRVACT(R1)           ;SET THE DRIVE ACTIVE INDICATOR
037656  105061  035634                          CLRB    DRVSTA(R1)              ;PUT DRIVE STATUS TO OFFLINE
037662  112761  000001  035702                  MOVB    #1,ULDFLG(R1)           ;SET ''UNLOAD IN PROGRESS'' FLAG
037670  010346                                  MOV     R3,-(SP)                ;START THE ''UNLOAD'' COMMAND
037672  004037  043076                          JSR     R0,WRT.RM
037676  000000                                  RMCS1
037700  040122                                  CI7
037702  000207                                  RTS     PC                      ;RETURN TO USER
037704  122703  000143                  5$:     CMPB    #143,R3                 ;IS IT A ''SET FORMAT'' COMMAND?
037710  001014                                  BNE     6$                      ;BR IF NO
037712  004037  042722                          JSR     R0,RD.RM                ;READ THE OFFSET REGISTER
037716  000032                                  RMOF
037720  040122                                  CI7
037722  116266  000001  000001                  MOVB    1(R2),1(SP)             ;COMBINE ''FMT16'',''ECI'', AND ''HCI''
037730  004037  043076                          JSR     R0,WRT.RM               ;LOAD ''FMT16'', ''ECI'', AND/OR ''HCI''.
037734  000032                                  RMOF
037736  040122                                  CI7
037740  000436                                  BR      12$
037742  122703  000141                  6$:     CMPB    #141,R3                 ;IS IT A ''GET REGISTER'' COMMAND?
037746  001023                                  BNE     10$                     ;BR IF NO
037750  016203  000006                  7$:     MOV     6(R2),R3                ;POINTS TO 1ST ADDRESS OF WHERE
                                                                                ;TO PUT THE REGISTER(S)
037754  116237  000010  037772                  MOVB    10(R2),9$               ;INIT. THE INDEX FOR THE FIRST REG.
037762  116205  000011                          MOVB    11(R2),R5               ;INDEX OF LAST REG. TO MOVE
037766  004037  042722                  8$:     JSR     R0,RD.RM                ;READ RH/RM REGISTER
037772  000000                          9$:     RMCS1                           ;INDEX OF REG. TO READ
037774  040122                                  CI7
037776  012623                                  MOV     (SP)+,(R3)+             ;GET THE CONTENTS OF RH/RM REG.
040000  023705  037772                          CMP     9$,R5                   ;LAST REG. BEEN READ?
040004  001414                                  BEQ     12$                     ;GET OUT IF YES
040006  062737  000002  037772                  ADD     #2,9$                   ;INCREASE THE INDEX BY 2
```

```
040014  000764                          BR      8$                ;LOOP--MORE TO READ
040016  122703  000145          10$:    CMPB    #145,R3           ;IS IT A ''SELECT DRIVE'' COMMAND?
040022  001405                          BEQ     12$               ;BR IF YES
040024  010346                  11$:    MOV     R3,-(SP)          ;LOAD THE COMMAND
040026  004037  043076                  JSR     R0,WRT.RM
040032  000000                          RMCS1
040034  040122                          CI7
040036  004737  044142          12$:    JSR     PC,POPQUE         ;REMOVE REQ. FROM QUEUE
040042  052762  000200  000016          BIS     #BIT07,16(R2)     ;SET THE 'DONE'' BIT
040050  005737  035722                  TST     SAVEFG            ;SAVE THE RH/RM REGISTERS?
040054  100002                          BPL     13$               ;BR IF NO
040056  004737  043264                  JSR     PC,SVRH70         ;YES--GO SAVE THE REGISTERS
040062  000207                  13$:    RTS     PC                ;RETURN TO USER
040064  006301                  CI5:    ASL     R1
040066  012761  060000  035726          MOV     #60000,TIMER(R1)          ;SET A ONE SECOND TIMER
040074  006201                          ASR     R1
040076  112761  000001  035624          MOVB    #1,DRVACT(R1)     ;SET THE DRIVE ACTIVE
040104  000207                          RTS     PC                ;RETURN TO THE USER
040106  010346                  CI6:    MOV     R3,-(SP)          ;LOAD THE COMMAND
040110  004037  043076                  JSR     R0,WRT.RM
040114  000000                          RMCS1
040116  040122                          CI7
040120  000761                          BR      CI5
040122  032764  010000  000010  CI7:    BIT     #BIT12,RMCS2(R4)          ;DRIVE NON-EXISTENT ?
                                ;       BNE     CI8               ;BR IF YES
040130  005702                  1$:     TST     R2                ;ANYTHING IN QUEUE ?
                                ;       BEQ     CI7B              ;BR IF NOT
040132  001001                          BNE     2$                ;BR IF QUEUE IS THERE
040134  000207                          RTS     PC                ;OTHERWISE EXIT
040136  012762  104000  000016  2$:     MOV     #BIT15!BIT11,16(R2)       ;SET ''PARITY'' ERROR INDICATOR
                                ;       JSR     PC,SVRH70         ;GO SAVE THE RH/RM REGISTERS
040144  012746  000111          CI7B:   MOV     #111,-(SP)        ;DO A 'DRIVE CLEAR''
040150  004037  043076                  JSR     R0,WRT.RM
040154  000000                          RMCS1
040156  040222                          CI8
040160  004737  044024          2$:     JSR     PC,EMPTYQ         ;EMPTY THE QUEUE
040164  105061  035664                  CLRB    DPRQS(R1)         ;CLEAR THE PORT REQUEST FLAG
040170  105061  035702                  CLRB    ULDFLG(R1)        ;CLEAR THE UNLOAD IN QUEUE FLAG
040174  105061  035624                  CLRB    DRVACT(R1)        ;DRIVE IS IDLE
040200  020237  035674                  CMP     R2,TRNSWT         ;IF THIS DRIVE HAD AN I/O REQUEST
                                ;       CMP     R1,DTUW           ;IF THIS DRIVE HAD AN I/O REQUEST
040204  001005                          BNE     1$                ;IN PROGRESS CLEAR ALL OF THE FLAGS
040206  005037  035674                  CLR     TRNSWT
040212  012737  177777  035746          MOV     #-1,DTUW
040220  000207                  1$:     RTS     PC
040222  104412                  CI8:    SAVREG                    ;SAVE R0 - R5
040224  032764  010000  000010          BIT     #BIT12,RMCS2(R4)          ;IS 'NED' SET ?
                                ;       BNE     1$                ;BR IF YES
040232  005001                          CLR     R1
040234  005003                          CLR     R3
040236  105761  035624          1$:     TSTB    DRVACT(R1)        ;DRIVE ACTIVE?
                                ;       BEQ     5$                ;BR IF NO
040242  001003                          BNE     22$               ;BR IF IN ACTIVE
040244  105761  035664                  TSTB    DPRQS(R1)         ;PORT REQUEST
040250  001443                          BEQ     5$                ;BR IF NOT
040252  013702  035674          22$:    MOV     TRNSWT,R2         ;GET THE ''TRANSFER WAIT'' QUEUE
040256  020137  035746                  CMP     R1,DTUW           ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
```

```
040262  001402                          BEQ     2$                  ;BR IF YES
040264  004737  044120                  JSR     PC,GETREQ           ;GET THE DPB POINTER
040270  005702                  2$:     TST     R2                  ;QUEUE ENTRY FOR DRIVE ?
040272  001413                          BEQ     4$                  ;BR IF NOT
040274  032764  010000  000010          BIT     #BIT12,RMCS2(R4)            ;'NED' SET ?
040302  001404                          BEQ     3$                  ;BR IF NOT
040304  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)         ;SET 'DRIVE NON-EXISTENT' INDICATOR
040312  000403                          BR      4$                  ;CONTINUE
040314  012762  102000  000016  3$:     MOV     #BIT15!BIT10,16(R2)         ;SET ''NON-CLEARABLE PARITY'' ERROR INDICATOR
                                ;       JSR     PC,SVRH70           ;SAVE RH/RM REGISTERS
040322  012763  177777  035726  4$:     MOV     #-1,TIMER(R3)       ;STOP THE TIMER
040330  105061  035624                  CLRB    DRVACT(R1)          ;SET ''DRIVE ACTIVE'' TO IDLE
040334  105061  035664                  CLRB    DPRQS(R1)           ;CLEAR PORT REQUEST FLAG
040340  020137  035746                  CMP     R1,DTUW             ;IS THIS DRIVE SETUP FOR A TRANSFER
040344  001005                          BNE     5$                  ;BR IF NOT
040346  012737  177777  035746          MOV     #-1,DTUW            ;RESET THE INDICATOR
040354  005037  035674                  CLR     TRNSWT              ;CLEAR THE TRANSFER QUEUE
040360  105061  035702          5$:     CLRB    ULDFLG(R1)          ;CLEAR UNLOAD FLAG
040364  032764  010000  000010          BIT     #BIT12,RMCS2(R4)            ;'NED' SET ?
                                ;       BNE     6$                  ;BR IF YES
040372  005201                          INC     R1                  ;MOVE TO THE NEXT DRIVE
040374  062703  000002                  ADD     #2,R3
040400  042701  177770                  BIC     #^C7,R1
040404  001314                          BNE     1$                  ;BR IF MORE DRIVES
040406  012737  177777  035746          MOV     #-1,DTUW            ;NO DATA TRANSFERS UNDERWAY
040414  005037  035674                  CLR     TRNSWT              ;CLEAR THE 'TRANSFER WAIT' QUEUE
040420  004737  043746                  JSR     PC,CLRQUE           ;CLEAR ALL OF THE REQUEST QUEUES
040424  012764  000040  000010          MOV     #CLR,RMCS2(R4)      ;DO A MASSBUS INIT.
040432  000406                          BR      7$                  ;CONTINUE
040434  004737  044024          6$:     JSR     PC,EMPTYQ           ;CLEAR THE DRIVE'S QUEUE
040440  105061  035634                  CLRB    DRVSTA(R1)          ;SET DRIVE TO OFFLINE
040444  105061  035644                  CLRB    DRVTYP(R1)          ;CLEAR THE DRIVE TYPE INDICATOR
040450  004737  043402          7$:     JSR     PC,SET.IE           ;SET ''IE'' WITHOUT ''TRE''
040454  104413                          RESREG                      ;RESTORE R0 - R5
040456  000207                          RTS     PC                  ;RETURN

                                ;LOOK AHEAD ROUTINE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1          ;DRIVE NUMBER
                                ;       MOV     #DPB,R2             ;POINT TO DPB
                                ;       JSR     R0,LA               ;GO CHECK THE WINDOW
                                ;       RETURN1                     ;ERROR RETURN
                                ;       RETURN2                     ;START A SEARCH
                                ;       RETURN3                     ;START A DATA TRANSFER

040460  013704  035762          LA:     MOV     RMADR,R4            ;GET RMCS1'S ADDRESS
040464  010164  000010                  MOV     R1,RMCS2(R4)        ;SELECT DRIVE
040470  004037  042722                  JSR     R0,RD.RM            ;READ DRIVE STATUS
040474  000012                          RMDS
040476  040626                          4$                          ;ERROR RETURN ADDRESS
040500  042716  157577                  BIC     #^C020200,(SP)      ;ON CYLINDER ?
040504  022726  000200                  CMP     #200,(SP)+          ;PIP=0,DRY=1?
040510  001044                          BNE     3$                  ;NO
040512  105261  035712                  INCB    LACNT(R1)           ;INCREMENT THE LOOK AHEAD COUNT
040516  126137  035712  035770          CMPB    LACNT(R1),MXLACT            ;EXCEED MAX?
040524  003033                          BGT     2$                  ;BR IF YES
```

```
040526  116203  000010                  MOVB    10(R2),R3               ;GET DESIRED SECTOR ADDRESS AND
040532  000303                          SWAB    R3                      ;MULT. BY 64--ALIGN WITH
040534  006203                          ASR     R3                      ;LOOK AHEAD REGISTER
040536  006203                          ASR     R3
040540  012737  000340  177776          MOV     #340,PS                 ;PRIORITY LEVEL '7''
040546  004037  042722          6$:     JSR     R0,RD.RM                ;READ LOOK AHEAD REGISTER
040552  000020                          RMLA
040554  040626                          4$
040556  021664  000020                  CMP     (SP),RMLA(R4)           ;CORRECT LA NUMBER ?
040562  001402                          BEQ     7$                      ;YES
040564  005726                          TST     (SP)+                   ;NO,CLEAR STACK
040566  000415                          BR      3$
040570  162603                  7$:     SUB     (SP)+,R3                ;CALCULATE THE DELTA
040572  002002                          BGE     1$
040574  062703  004000                  ADD     #<32.*64.>,R3           ;MAKE THE DELTA POSITIVE
040600  023703  035772          1$:     CMP     MXDLTA,R3               ;CHECK THE DELTA TO SEE
040604  002406                          BLT     3$                      ;IF IT IS WITHIN THE
040606  023703  035774                  CMP     MNDLTA,R3               ;WINDOW---IF YES, ZERO
040612  002003                          BGE     3$                      ;THE LOOK AHEAD COUNT
040614  105061  035712          2$:     CLRB    LACNT(R1)               ;AND TAKE THE I/O EXIT
040620  005720                          TST     (R0)+
040622  005720                  3$:     TST     (R0)+                   ;ADJUST THE RETURN ADDRESS
040624  000402                          BR      5$                      ;EXIT
040626  004737  040122          4$:     JSR     PC,CI7                  ;PROCESS THE ERROR
040632  000200                  5$:     RTS     R0                      ;RETURN

                                ;INTERRUPT SERVICE ROUTINE

040634  112737  000001  035700  ISR:    MOVB    #1,ACTDRV               ;SET ''ACTIVE DRIVER'' FLAG
040642  104412                          SAVREG                          ;SAVE R0 - R5
040644  013704  035762                  MOV     RMADR,R4                ;ADDRESS OF RHSCS1
040650  013701  035746                  MOV     DTUW,R1                 ;GET ''DATA TRANSFER UNDERWAY'' INDICATOR
040654  002403                          BLT     1$                      ;BR IF NO DATA TRANSFER UNDERWAY
040656  004737  040700                  JSR     PC,TD                   ;CALL TRANSFER DONE
040662  000402                          BR      2$                      ;EXIT
040664  004737  041154          1$:     JSR     PC,SC                   ;CALL SPECIAL CONDITIONS
040670  104413                  2$:     RESREG                          ;RESTORE R0 - R5
040672  105037  035700                  CLRB    ACTDRV                  ;CLEAR ''ACTIVE DRIVER'' FLAG
040676  000002                          RTI                             ;RETURN

                                ;TRANSFER DONE ROUTINE

040700  105061  035624          TD:     CLRB    DRVACT(R1)              ;SET DRIVE ACTIVE INDICATOR TO IDLE
040704  012737  177777  035746          MOV     #-1,DTUW                ;NO DATA TRANSFERS UNDERWAY
040712  006301                          ASL     R1
040714  012761  177777  035726          MOV     #-1,TIMER(R1)           ;CANCEL TIMEOUT
040722  006201                          ASR     R1
040724  013702  035674                  MOV     TRNSWT,R2               ;GET ''DPB'' ADDRESS FROM THE
040730  005037  035674                  CLR     TRNSWT                  ;TRANSFER WAIT QUEUE--CLEAR QUEUE
040734  052762  000200  000016          BIS     #BIT07,16(R2)           ;SET DONE
040742  010164  000010                  MOV     R1,RMCS2(R4)            ;SELECT THE DRIVE
040746  004037  042722                  JSR     R0,RD.RM                ;TRANSFER ERROR(TRE=1)?
040752  000000                          RMCS1
040754  040122                          CI7
040756  006126                          ROL     (SP)+
040760  100421                          BMI     3$                      ;BR IF YES
040762  005737  035722                  TST     SAVEFG                  ;SAVE THE RH/RM REGISTERS?
```

```
040766  100002                           BPL     1$                      ;BR IF NO
040770  004737  043264                   JSR     PC,SVRH70               ;YES--SAVE THE REGISTERS
040774  004737  041054          1$:      JSR     PC,WC.HK                        ;SEE IF WRITE CHECK TO BE PUT IN QUEUE
041000  004737  044120                   JSR     PC,GETREQ               ;GET DPB POINTER
041004  005702                           TST     R2                      ;ENTRY FOR DRIVE ?
041006  001403                           BEQ     2$                      ;BR IF NOT
041010  004737  037042                   JSR     PC,OPT                  ;CALL OPTIMIZER
041014  000457                           BR      SC                      ;CHECK OTHER DRIVES

                                ;THE RELEASE DRIVE COMMAND IS FORECD TO ENTER FOR DUAL PORT OPERATION
041016  012714  000113          2$:      MOV     #113,(R4)               ;RELEASE THE DRIVE
041022  000454                           BR      SC                      ;CHECK FOR OTHER DRIVES
041024  052762  100100  000016  3$:      BIS     #BIT15!BIT06,16(R2)             ;SET DATA ERROR FLAG
041032  004737  044024                   JSR     PC,EMPTYQ               ;EMPTY THE ''DRIVE'S WAIT'' QUEUE
041036  004737  043264                   JSR     PC,SVRH70               ;SAVE THE RH/RM REGISTERS
041042  012714  040111                   MOV     #40111,(R4)             ;ISSUE A ''DRIVE CLEAR''
041046  012714  000113                   MOV     #113,(R4)               ;ISSUE A RELEASE TO THE DRIVE
041052  000440                           BR      SC                      ;CHECK FOR OTHER DRIVES


041054  122762  000002  000024  WC.HK:   CMPB    #2,$CODE(R2)            ;LAST OPERATION WRITE DATA ?
041062  001404                           BEQ     1$                      ;BR IF IT WAS
041064  122762  000003  000024           CMPB    #3,$CODE(R2)            ;LAST OPERATION WRITE HEADER & DATA ?
041072  001027                           BNE     2$                      ;BR IF NOT
041074  004037  044044          1$:      JSR     R0,DRVQUE               ;PUT THE OPERATION IN THE QUEUE
041100  000424                           BR      2$                      ;QUEUE IS FULL
041102  005062  000016                   CLR     16(R2)                  ;CLEAR 'DONE' BIT IN DPB
041106  116262  002116  000027           MOVB    $RMCS1(R2),$PREVO(R2)           ;SAVE WRITE OPERATION CODE
041114  016262  000012  000034           MOV     $CYL(R2),$PREVA+2(R2)           ;SAVE CYLINDER
041122  016262  000010  000032           MOV     $SEC(R2),$PREVA(R2)             ;SAVE SECTOR AND TRACK ADDRESSES
041130  142762  000002  000024           BICB    #2,$CODE(R2)            ;CHANGE WRITE TO CHECK
041136  142762  000020  000002           BICB    #20,$COMND(R2)          ;CHANGE DRIVER CODE TO WRITE CHECK
041144  152762  000010  000002           BISB    #10,$COMND(R2)          ;FINISH CHANGING CODE TO WRITE CHECK
041152  000207                  2$:      RTS     PC                      ;EXIT

                                ;SPECIAL CONDITION ROUTINE

041154  116403  000016          SC:      MOVB    RMAS(R4),R3             ;READ ''RMAS''
041160  001012                           BNE     2$                      ;BR IF ANY 'ATA' BITS SET
041162  004037  042722                   JSR     R0,RD.RM                ;READ CONTROL AND STATUS REGISTER
041166  000000                           RMCS1
041170  040222                           CI8
                                ;        1$                              ;EXIT IF FAIL TO READ
041172  106126                           ROLB    (SP)+                   ;IS ''IE''=1?
041174  100403                           BMI     1$                      ;YES, NO DRIVES TO CHECK
041176  104001                           EMT     1                       ;REPORT AN ILLEGAL INTERRUPT
041200  004737  043402                   JSR     PC,SET.IE               ;SET INTERRUPT ENABLE
041204  000207                  1$:      RTS     PC                      ;RETURN
041206  005046                  2$:      CLR     -(SP)                   ;PROCESS ALL DRIVES THAT HAVE
041210  110316                           MOVB    R3,(SP)                 ;AN ''ATA''=1
041212  012703  000001                   MOV     #1,R3
041216  005001                           CLR     R1
041220  030316                  SC3:     BIT     R3,(SP)                 ;ATA=1?
041222  001005                           BNE     SC5                     ;YES
041224  005201                  SC4:     INC     R1                      ;MOVE TO THE NEXT DRIVE
041226  106303                           ASLB    R3
041230  001373                           BNE     SC3                     ;BR IF MORE TO CHECK?
041232  005726                           TST     (SP)+                   ;CLEAN OFF THE STACK
```

```
041234  000207                                  RTS     PC                      ;RETURN TO USER
041236  105761  035654                  SC5:    TSTB    DPINT(R1)               ;INITIALIZING THE DRIVE ?
041242  001402                                  BEQ     1$                      ;BR IF NOT
041244  000137  042152                          JMP     SC13                    ;PROCESS THE DRIVE
041250  105761  035664                  1$:     TSTB    DPRQS(R1)               ;PORT REQUEST OUTSTANDING ?
041254  001402                                  BEQ     2$                      ;BR IF NOT
041256  000137  042152                          JMP     SC13                    ;START THE OUTSTANDING COMMAND
041262  105761  035634                  2$:     TSTB    DRVSTA(R1)              ;CHECK THE DRIVE STATUS
041266  003023                                  BGT     5$                      ;BR IF ONLINE
041270  105761  035702                          TSTB    ULDFLG(R1)              ;UNLOAD IN PROGRESS?
041274  003420                                  BLE     5$                      ;BR IF NOT
041276  004737  044120                          JSR     PC,GETREQ               ;GET DPB POINTER
041302  004737  043264                          JSR     PC,SVRH70               ;SAVE THE RH/RM REGISTERS
041306  004737  042102                          JSR     PC,SC12                 ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                                ;ALSO DO A DRIVE INIT (DRVINT)
041312  105761  035634                          TSTB    DRVSTA(R1)              ;DID DRIVE COME ONLINE?
041316  003414                                  BLE     6$                      ;NO
041320  032737  040000  035614                  BIT     #BIT14,RMERRS           ;WAS THERE AN ERROR?
041326  001000                                  BNE     3$                      ;BR IF ERROR
                                        ;       JMP     SC11                    ;NO ERROR
041330  013705  035616                  3$:     MOV     RMERRS+2,R5             ;YES -- PICKUP RMER1 AND
041334  000500                                  BR      SC6A                    ;GO PROCESS THE ERROR
041336  105761  035624                  5$:     TSTB    DRVACT(R1)              ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
041342  001027                                  BNE     SC6                     ;BR IF EITHER
041344  004737  042102                          JSR     PC,SC12                 ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                                ;ALSO DO A DRVINT
041350  105761  035654                  6$:     TSTB    DPINT(R1)               ;TRYING TO INIT THE DRIVE ?
041354  001323                                  BNE     SC4                     ;BR IF YES, CHECK ON MORE DRIVES
041356  105761  035634                          TSTB    DRVSTA(R1)              ;CHECK ON DRIVE'S STATUS
041362  100412                                  BMI     7$                      ;BR IF UNSAFE
041364  032737  020000  035620                  BIT     #BIT13,RMERRS+4         ;ADDRESS PLUG CHANGED ?
041372  001011                                  BNE     8$                      ;BR IF YES
                                        ;       MOV     #113,-(SP)              ;RELEASE COMMAND
041374  012746  000111                          MOV     #111,-(SP)              ;DRIVE CLEAR
041400  004037  043076                          JSR     R0,WRT.RM               ;WRITE THE COMMAND INTO RMCS1
041404  000000                                  RMCS1                           ;REGISTER INDEX
041406  041742                                  SC8                             ;PARITY EXIT ADDRESS
041410  011605                          7$:     MOV     (SP),R5                 ;PICKUP (RMAS) BEFORE THE ERROR CALL
041412  104002                                  EMT     2                       ;REPORT THE UNEXPECTED ATTENTION
041414  000703                                  BR      SC4                     ;GO CHECK FOR MORE ATA'S
041416                                  8$:
041416  104005                                  EMT     5                       ;REPORT THE ADDRESS PLUG CHANGE
041420  000701                                  BR      SC4                     ;CHECK FOR MORE DRIVES
041422  006301                          SC6:    ASL     R1                      ;SETUP TO ADDRESS WORDS
041424  012761  177777  035726                  MOV     #-1,TIMER(R1)           ;STOP THE TIMER
041432  006201                                  ASR     R1                      ;RESTORE THE DRIVE ADDRESS
041434  004737  044120                          JSR     PC,GETREQ               ;GET THE DPB POINTER FROM THE QUEUE
041440  010164  000010                          MOV     R1,RMCS2(R4)            ;SELECT DRIVE
041444  000137  041772                          JMP     SC11                    ;PROCESS THE SEARCH
041450  004037  042722                          JSR     R0,RD.RM                ;READ THE RM'S STATUS REG.
041454  000012                                  RMDS
041456  041742                                  SC8
041460  011605                                  MOV     (SP),R5                 ;AND PUT IT IN R5
041462  006126                                  ROL     (SP)+                   ;WAS THERE AN ERROR?
041464  100407                                  BMI     1$                      ;BR IF ERROR
041466  105761  035624                          TSTB    DRVACT(R1)              ;CHECK DRIVE'S STATE
041472  003137                                  BGT     SC11                    ;BR IF DRIVE ACTIVE WITH ORDER
```

```
    886 041474  052762  100210  000016           BIS     #BIT15!BIT07!BIT03,16(R2)          ;INFORM USER OF ERROR RECOVER COMPLETION
    887 041502  000470                           BR      SC7
    888 041504  004037  042722          1$:      JSR     R0,RD.RM            ;READ ERROR REGISTER #1
    889 041510  000014                           RMER1
    890 041512  041742                           SC8
    891 041514  012605                           MOV     (SP)+,R5            ;AND SAVE IT IN R5
    892 041516  004737  043264                   JSR     PC,SVRH70           ;SAVE RH/RM REGISTERS
    893 041522  012746  000111                   MOV     #111,-(SP)          ;ISSUE A DRIVE CLEAR
    894 041526  004037  043076                   JSR     R0,WRT.RM
    895 041532  000000                           RMCS1
    896 041534  041742                           SC8
    897 041536  006105                  SC6A:    ROL     R5                  ;WAS 'UNSAFE'' CONDITION =1?
    898 041540  100406                           BMI     1$                  ;BR IF YES
    899 041542  005702                           TST     R2                  ;ANYTHING IN QUEUE ?
    900 041544  001447                           BEQ     SC7                 ;BR IF NOT
    901 041546  052762  100240  000016           BIS     #BIT15!BIT07!BIT05,16(R2)          ;INFORM USER OF ERROR
    902 041554  000443                           BR      SC7
    903 041556  004037  042722          1$:      JSR     R0,RD.RM            ;READ DRIVE STATUS REG. #1
    904 041562  000012                           RMDS
    905 041564  041742                           SC8
    906 041566  011605                           MOV     (SP),R5             ;SAVE RMDS IN R5
    907 041570  006126                           ROL     (SP)+               ;'ERR''=1?
    908 041572  100011                           BPL     2$                  ;BR IF NO--UNSAFE CLEARED
    909 041574  112761  177777  035634           MOVB    #-1,DRVSTA(R1)      ;DRIVE IS UNSAFE
    910 041602  004737  043264                   JSR     PC,SVRH70           ;SAVE RH/RM REGISTERS
    911 041606  052762  110000  000016           BIS     #BIT15!BIT12,16(R2)        ;INFORM USER OF UNSAFE ERROR
    912 041614  000423                           BR      SC7
    913 041616  032705  010000          2$:      BIT     #BIT12,R5           ;'MOL'' = 1 ?
    914 041622  001015                           BNE     3$                  ;BR IF YES
    915 041624  112761  177777  035624           MOVB    #-1,DRVACT(R1)      ;ACTIVE ERROR RECOVER
    916 041632  112761  000001  035634           MOVB    #1,DRVSTA(R1)       ;ONLINE
    917 041640  006301                           ASL     R1
    918 041642  012761  072460  035726           MOV     #30000.,TIMER(R1)          ;START 30 SECOND TIMER
    919 041650  006201                           ASR     R1
    920 041652  000137  041224                   JMP     SC4
    921 041656  052762  100220  000016  3$:      BIS     #BIT15!BIT07!BIT04,16(R2)          ;INFORM USER OF ERROR
    922 041664  105061  035624          SC7:     CLRB    DRVACT(R1)          ;DRIVE IS IDLE
    923                                 ;        JSR     PC,EMPTYQ           ;DUMP THE QUEUE
    924 041670  004737  044142                   JSR     PC,POPQUE           ;REMOVE THE QUEUE
    925 041674  105761  035702                   TSTB    ULDFLG(R1)          ;UNLOAD IN RMOGRESS OR QUEUE?
    926 041700  003002                           BGT     1$                  ;BR IF NOT
    927 041702  105061  035702                   CLRB    ULDFLG(R1)          ;CLEAR UNLOAD FLAG
    928 041706  116164  035750  000016  1$:      MOVB    ATABIT(R1),RMAS(R4)        ;CLEAR ATTENTION BIT
    929 041714  105761  035634                   TSTB    DRVSTA(R1)          ;IS THE DRIVE UNSAFE ?
    930 041720  100406                           BMI     2$                  ;BR IF IT IS
    931                                 ;        MOV     #113,-(SP)          ;RELEASE COMMAND
    932 041722  012746  000111                   MOV     #111,-(SP)          ;DRIVE CLEAR COMMAND
    933 041726  004037  043076                   JSR     R0,WRT.RM           ;WRITE THE COMMAND INTO RPCS1
    934 041732  000000                           RMCS1                       ;REGISTER INDEX
    935 041734  041742                           SC8                         ;PARITY EXIT ADDRESS
    936 041736  000137  041224          2$:      JMP     SC4                 ;CHECK FOR MORE DRIVES
    937 041742  105761  035624          SC8:     TSTB    DRVACT(R1)          ;IS DRIVE IDLE?
    938 041746  001405                           BEQ     1$                  ;YES
    939 041750  004737  044120                   JSR     PC,GETREQ           ;GET DPB POINTER
    940 041754  004737  040122                   JSR     PC,CI7              ;PROCESS THE PARITY ERROR
    941 041760  000402                           BR      2$                  ;CONTINUE
    942 041762                          1$:
```

```
    943                                     ;       JSR     PC,CI7                  ;PROCESS THE PARITY ERROR
    944 041762  004737  040144                      JSR     PC,CI7B                 ;PROCESS THE UNCORRECTABLE PARITY ERROR
    945 041766  000137  041224              2$:     JMP     SC4                     ;CHECK MORE DRIVES
    946 041772  105761  035702              SC11:   TSTB    ULDFLG(R1)              ;''UNLOAD IN PROGRESS''?
    947 041776  003402                              BLE     1$                      ;BR IF NO
    948 042000  105061  035702                      CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
    949 042004  105061  035624              1$:     CLRB    DRVACT(R1)              ;SET DRIVE IDLE
    950 042010  136137  035750  035676              BITB    ATABIT(R1),SRCHWT               ;DOING A SEARCH OPERATION FOR
    951                                                                                     ;AN I/O COMMAND?
    952 042016  001012                              BNE     2$                      ;BR IF YES
    953 042020  004737  044142                      JSR     PC,POPQUE               ;REMOVE REQUEST FROM QUEUE
    954 042024  052762  000200  000016              BIS     #BIT07,16(R2)           ;SET ''DONE'' BIT
    955 042032  005737  035722                      TST     SAVEFG                  ;SAVE THE REGISTERS?
    956 042036  100002                              BPL     2$                      ;BR IF NO
    957 042040  004737  043264                      JSR     PC,SVRH70               ;YES--SAVE ALL OF THE RH/RM REG'S
    958 042044  116164  035750  000016      2$:     MOVB    ATABIT(R1),RMAS(R4)             ;CLEAR ATTENTION BIT
    959 042052  146137  035750  035676              BICB    ATABIT(R1),SRCHWT               ;CLEAR IMPLIED SEEK SET
    960 042060  006301                              ASL     R1                      ;WORD INDEX
    961 042062  012761  177777  035726              MOV     #-1,TIMER(R1)           ;STOP CLOCK
    962 042070  006201                              ASR     R1                      ;RESTORE R1
    963 042072  004737  037042                      JSR     PC,OPT                  ;START A REQUEST
    964 042076  000137  041224                      JMP     SC4                     ;CHECK FOR MORE DRIVES
    965 042102  010164  000010              SC12:   MOV     R1,RMCS2(R4)            ;SELECT DRIVE
    966 042106  016437  000012  035614              MOV     RMDS(R4),RMERRS         ;SAVE THE FOUR REGISTERS THAT
    967 042114  016437  000014  035616              MOV     RMER1(R4),RMERRS+2              ;WILL TELL US SOMETHING
    968 042122  016437  000042  035620              MOV     RMER2(R4),RMERRS+4
    969 042130  016437  000040  035622              MOV     RMMR2(R4),RMERRS+6
    970 042136  004037  036212                      JSR     R0,DRVINT               ;INIT. THE STATE OF THE DRIVE
    971 042142  000401                              BR      1$                      ;TAKE ERROR EXIT
    972 042144  000207                              RTS     PC                      ;RETURN
    973 042146  005726                      1$:     TST     (SP)+                   ;POP PC OFF OF THE STACK
    974 042150  000674                              BR      SC8                     ;PROCESS THE PARITY ERROR
    975 042152  006301                      SC13:   ASL     R1                      ;SETUP TO ADDRESS WORDS
    976 042154  012761  177777  035726              MOV     #-1,TIMER(R1)           ;STOP THE TIMER
    977 042162  006201                              ASR     R1                      ;
    978 042164  010164  000010                      MOV     R1,RMCS2(R4)            ;SELECT THE DRIVE
    979 042170  116164  035750  000016              MOVB    ATABIT(R1),RMAS(R4)             ;CLEAR THE ATTENTION BIT
    980 042176  105761  035654              1$:     TSTB    DPINT(R1)               ;INITIALIZING THE DRIVE ?
    981 042202  001424                              BEQ     2$                      ;BR IF NOT
    982 042204  105061  035654                      CLRB    DPINT(R1)               ;CLEAR THE INIT INDICATOR
    983 042210  004037  036212                      JSR     R0,DRVINT               ;GO INIT THE DRIVE
    984 042214  000240                              NOP                             ;DUMMY PARITY ERROR RETURN
    985 042216  105761  035634                      TSTB    DRVSTA(R1)              ;DRIVE ONLINE ?
    986 042222  003014                              BGT     2$                      ;BR IF YES -- START ORDER
    987 042224  005702                              TST     R2                      ;QUEUE ENTRY FOR THE DRIVE
    988 042226  001426                              BEQ     3$                      ;BR IF NOT
    989 042230  004737  044120                      JSR     PC,GETREQ               ;GET DPB ADDRESS
    990 042234  052762  140000  000016              BIS     #BIT15!BIT14,16(R2)             ;INFORM USER THAT DRIVE OFFLINE
    991 042242  004737  043264                      JSR     PC,SVRH70               ;SAVE THE REGISTERS
    992                                     ;       JSR     PC,EMPTYQ               ;EMPTY THE REQUEST QUEUE
    993 042246  004737  044142                      JSR     PC,POPQUE               ;REMOVE THE QUEUE
    994 042252  000414                              BR      3$
    995 042254  032764  004000  000000      2$:     BIT     #BIT11,RMCS1(R4)                ;DVA SET ?
    996 042262  001006                              BNE     4$                      ;SET THEN CALL OPT
    997 042264  006301                              ASL     R1
    998 042266  012761  060000  035726              MOV     #60000,TIMER(R1)
    999 042274  006201                              ASR     R1
```

```
042276  000402                            BR      3$
042300  004737  037042            4$:     JSR     PC,OPT           ;START THE PENDING REQUEST
042304  000137  041224            3$:     JMP     SC4              ;PROCESS OTHER DRIVES

                                  ;RM TIMER ROUTINE
                                  ;CALL
                                  ;       MOV     #TIME,-(SP)      ;ELASPED TIME IN MILLISECONDS ON THE STACK
                                  ;       JSR     PC,RMTMR         ;CALL RM05 TIME ROUTINE

042310  005737  035700            RMTMR:  TST     ACTDRV           ;CHECK ''ACTDRV & ACTSTR''
042314  001027                            BNE     4$               ;IF NON ZERO EXIT
042316  112737  000001  035701            MOVB    #1,ACTSTR        ;SET ''ACTSTR''
042324  104412                            SAVREG                   ;SAVE R0 - R5
042326  005001                            CLR     R1               ;START WITH DRIVE 0
042330  005003                            CLR     R3
042332  005763  035726            1$:     TST     TIMER(R3)        ;IS THE TIMER RUNNING?
042336  002406                            BLT     2$               ;BR IF NO
042340  166663  000002  035726            SUB     2(SP),TIMER(R3)  ;COUNT THE INTERVAL
042346  003002                            BGT     2$               ;BR IF NO SOFTWARE TIMEOUT
042350  004737  042400                    JSR     PC,STO           ;CALL SOFTWARE TIMEOUT ROUTINE
042354  005201                    2$:     INC     R1               ;MOVE TO NEXT DRIVE
042356  005723                            TST     (R3)+
042360  022701  000010                    CMP     #8.,R1           ;OUT OF DRIVES?
042364  003362                            BGT     1$               ;BR IF NO
042366  104413                    3$:     RESREG                   ;RESTORE R0 - R5
042370  105037  035701                    CLRB    ACTSTR           ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
042374  012616                    4$:     MOV     (SP)+,(SP)       ;ADJUST THE STACK
042376  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

042400  010146                    STO:    MOV     R1,-(SP)         ;SAVE R1
042402  010246                            MOV     R2,-(SP)         ;SAVE R2
042404  010346                            MOV     R3,-(SP)         ;SAVE R3
042406  010446                            MOV     R4,-(SP)         ;SAVE R4
042410  013704  035762                    MOV     RMADR,R4         ;GET ADDRESS OF ''RMCS1''
042414  010164  000010                    MOV     R1,RMCS2(R4)     ;SELECT THE DRIVE
042420  004037  042722                    JSR     R0,RD.RM         ;READ ''DRIVE STATUS REG''
042424  000012                            RMDS
                                  ;       STO5
042426  042710                            STO9
042430  105726                            TSTB    (SP)+            ;IS ''DRY''=1?
042432  100436                            BMI     STO2             ;BR IF YES
042434  105761  035654            STO1:   TSTB    DPINT(R1)        ;TRYING TO INTIALIZE THE DRIVE ?
042440  001033                            BNE     STO2             ;BR IF YES
042442  105761  035664                    TSTB    DPRQS(R1)        ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
042446  001030                            BNE     STO2             ;BR IF YES
042450  013702  035674                    MOV     TRNSWT,R2        ;PICKUP TRANSFER WAIT QUEUE
042454  020137  035746                    CMP     R1,DTUW          ;TRANSFER UNDERWAY ON THIS DRIVE?
```

```
042460  001404                          BEQ     1$                      ;BR IF YES
042462  000137  042710                  JMP     ST09                    ;IF NOT DON'T BOTHER DRIVES
042466  004737  044120                  JSR     PC,GETREQ               ;GET DPB ADDRESS
042472  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2)     ;SET THE ERROR FLAGS
042500  004737  043264                  JSR     PC,SVRH70               ;SAVE RH/RM REGISTERS
                                ;       MOV     #CLR,RMCS2(R4)          ;''INIT'' THE MASS BUS
042504  105061  035624                  CLRB    DRVACT(R1)              ;DRIVE IS IDLE
042510  105061  035702                  CLRB    ULDFLG(R1)              ;CLEAR THE UNLOAD FLAG
042514  005037  035674                  CLR     TRNSWT                  ;CLEAR DPB ADDRESS
042520  012737  177777  035746          MOV     #-1,DTUW                ;CLEAR THE TRANSFER DRIVE #
042526  000470                          BR      ST09                    ;DON'T BOTHER OTHER DRIVES
042530  116405  000016          ST02:   MOVB    RMAS(R4),R5             ;READ ATTENTION REG
042534  136105  035750                  BITB    ATABIT(R1),R5           ;IS ATTENTION FOR THIS DRIVE UP ?
042540  001007                          BNE     ST03                    ;YES
042542  105761  035654                  TSTB    DPINT(R1)               ;TRYING TO INTIALIZE THE DRIVE ?
042546  001021                          BNE     ST06                    ;BR IF YES - DRIVE NOT ONLINE
042550  105761  035664                  TSTB    DPRQS(R1)               ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
042554  001035                          BNE     ST07                    ;BR IF YES - NO RESPONSE TO REQUEST
042556  000454                          BR      ST09                    ;OTHER WISE EXIT
042560  105761  035654          ST03:   TSTB    DPINT(R1)               ;INITIALIZING THE DRIVE ?
042564  001003                          BNE     1$                      ;BR IF INIT PENDING
042566  105761  035664                  TSTB    DPRQS(R1)               ;PORT REQUEST PENDING ?
042572  001446                          BEQ     ST09                    ;BR IF NOT
042574  012763  177777  035726  1$:     MOV     #-1,TIMER(R3)           ;STOP THE TIMER
042602  000442                          BR      ST09                    ;EXIT
042604  004737  040222          ST05:   JSR     PC,CI8                  ;GO HANDLE THE PARITY ERROR
042610  000437                          BR      ST09
042612  105061  035654          ST06:   CLRB    DPINT(R1)               ;CLEAR THE INITIALIZE INDICATOR
042616  105061  035634                  CLRB    DRVSTA(R1)              ;SET UNIT OFFLINE
042622  012763  177777  035726          MOV     #-1,TIMER(R3)           ;STOP THE TIMER
042630  004737  044120                  JSR     PC,GETREQ               ;GET THE DPB ADDRESS
042634  005702                          TST     R2                      ;REQUEST IN QUEUE ?
042636  001424                          BEQ     ST09                    ;BR IF NOT
042640  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)     ;INFORM THE USER DRIVE NOT AVAILABLE
042646  000414                          BR      ST08                    ;FINISH
042650  012763  177777  035726  ST07:   MOV     #-1,TIMER(R3)           ;STOP THE TIMER
042656  105061  035664                  CLRB    DPRQS(R1)               ;CLEAR PORT REQUEST INDICATOR
042662  004737  044120                  JSR     PC,GETREQ               ;GET DPB ADDRESS
042666  005702                          TST     R2                      ;QUEUE ENTRY FOR DRIVE ?
042670  001407                          BEQ     ST09                    ;BR IF NONE
042672  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)      ;INFORM USER OF PORT REQUEST ERROR
042700  004737  044024          ST08:   JSR     PC,EMPTYQ               ;CLEAR THE QUEUE FOR THE DRIVE
042704  004737  043264                  JSR     PC,SVRH70               ;SAVE THE REGISTERS
042710  012604                  ST09:   MOV     (SP)+,R4                ;RESTORE R4
042712  012603                          MOV     (SP)+,R3                ;RESTORE R3
042714  012602                          MOV     (SP)+,R2                ;RESTORE R2
042716  012601                          MOV     (SP)+,R1                ;RESTORE R1
042720  000207                          RTS     PC                      ;RETURN

                                ;ROUTINE TO READ A RH/RM REGISTER
                                ;
                                ;CALL
                                ;       JSR     R0,RD.RM                ;GO READ A REGISTER
                                ;       INDEX                           ;REG. INDEX FROM BASE
                                ;       ERRADR                          ;ERROR ADDRESS--PROCESS ERROR STARTING
                                ;                                       ;AT THIS ADDRESS
                                ;       RETURN                          ;CONTENTS OF REG. IS ON THE STACK
```

```
                                        ;
042722  013737  035760  043064  RD.RM:  MOV     MCPEMX,RD.RM2       ;MAX. RETRYS ALLOWED
042730  011646                          MOV     (SP),-(SP)          ;SAVE R0 FOR RETURN
042732  013737  035762  042746          MOV     RMADR,RD.ADR        ;FORM THE DESIRED ADDRESS
042740  062037  042746                  ADD     (R0)+,RD.ADR        ;USING THE BASE AND THE INDEX
042744  013727                  RD.RM1: MOV     @(PC)+,(PC)+        ;READ THE DESIRED REGISTER OF THE RM DRIVE
042746  000000                  RD.ADR: .WORD   0                   ;ADDRESS IS FORMED HERE
042750  000000                  RD.WRD: .WORD   0                   ;REG. CONTENTS PUT HERE
042752  013766  042750  000002          MOV     RD.WRD,2(SP)        ;RETURN IT TO THE USER
042760  013746  035762                  MOV     RMADR,-(SP)         ;PUT THE ADDRESS ON THE STACK
042764  062716  000010                  ADD     #RMCS2,(SP)         ;FORM THE ADDRESS OF RMCS2
042770  032736  010000                  BIT     #BIT12,@(SP)+       ;CHECK THE 'NED' BIT
042774  001035                          BNE     RD.RM3              ;BR IF DRIVE NON-EXISTENT
042776  017746  172760                  MOV     @RMADR,-(SP)        ;READ RMCS1
043002  032716  020000                  BIT     #BIT13,(SP)         ;DID MCPE SET?
043006  001002                          BNE     1$                  ;BR IF YES
043010  022620                          CMP     (SP)+,(R0)+         ;ADJUST FOR RETURN
043012  000430                          BR      RD.RM4              ;EXIT
043014                          1$:
043014  104003                          EMT     3                   ;REPORT 'MCPE'' ERROR
043016  005737  035746                  TST     DTUW                ;DATA TRANSFER UNDERWAY?
043022  100405                          BMI     2$                  ;NO
043024  032716  040000                  BIT     #BIT14,(SP)         ;''TRE'' = 1 ?
043030  001402                          BEQ     2$                  ;NO
043032  005726                          TST     (SP)+               ;YES--CLEAN OFF THE STACK AND
043034  000415                          BR      RD.RM3              ;TAKE THE FATAL ERROR EXIT
043036  052716  040000          2$:     BIS     #BIT14,(SP)         ;CLEAR 'MCPE'' BY SENDING A ''1'' TO ''TRE''
043042  000316                          SWAB    (SP)                ;POSITION BEFORE WRITING
043044  013737  035762  043060          MOV     RMADR,3$            ;FORM ADDRESS OF HIGH BYTE
043052  005237  043060                  INC     3$
043056  112637                          MOVB    (SP)+,@(PC)+        ;WRITE THE HIGH BYTE OF RMCS1
043060  000000                  3$:     .WORD   0                   ;ADDRESS STORAGE
043062  005327                          DEC     (PC)+               ;EXCEEDED MAX. RETRYS
043064  000003                  RD.RM2: .WORD   3
043066  002326                          BGE     RD.RM1              ;BR IF NO
043070  011000                  RD.RM3: MOV     (R0),R0             ;FATAL ERROR EXIT
043072  012616                          MOV     (SP)+,(SP)
043074  000200                  RD.RM4: RTS     R0

                                ;ROUTINE TO WRITE A REGISTER
                                ;
                                ;CALL
                                ;               MOV     DATA,-(SP)          ;DATA TO BE LOADED ON THE STACK
                                ;               JSR     R0,WRT.RM           ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
                                ;               INDEX                       ;INDEX OF THE REGISTER TO BE LOADED
                                ;               ERRADR                      ;ADDRESS TO RETURN TO ON AN ERROR
                                ;               RETURN                      ;ERROR FREE RETURN

043076  013737  035760  043250  WRT.RM: MOV     MCPEMX,WRT.R2       ;MAX RETRYS ALLOWED
043104  016637  000002  043164          MOV     2(SP),WRT.WD        ;SAVE THE WORD TO WRITE
043112  012616                          MOV     (SP)+,(SP)          ;ADJUST THE STACK
043114  012037  043166                  MOV     (R0)+,WRT.AD        ;GET INDEX OF REGISTER TO BE WRITTEN
043120  001015                          BNE     1$                  ;BR IF NOT RMCS1
043122  122737  000150  043164          CMPB    #150,WRT.WD         ;IS THE COMMAND FOR DATA TRANSFERS?
043130  002411                          BLT     1$                  ;YES--DON'T GET THE OLD A16 & A17, & PSEL
043132  004037  042722                  JSR     R0,RD.RM            ;NO---COMBINE A16&A17, & PSEL WITH
043136  000000                          RMCS1                       ;THE COMMAND BEFORE SENDING IT TO
```

```
043140  043254                            WRT.R3                       ;THE RH/RM
043142  000316                            SWAB    (SP)
043144  042716  177770                    BIC     #^C7,(SP)
043150  112637  043165                    MOVB    (SP)+,WRT.WD+1
043154  063737  035762  043166  1$:       ADD     RMADR,WRT.AD         ;FORM THE ADDRESS OF THE DISK REG.
043162  012737                  WRT.R1:   MOV     (PC)+,@(PC)+         ;LOAD THE DESIRED REG.
043164  000000                  WRT.WD:   .WORD   0                    ;WORD TO WRITE GOES HERE
043166  000000                  WRT.AD:   .WORD   0                    ;ADDRESS IS FORMED HERE
043170  013746  035762                    MOV     RMADR,-(SP)          ;PUT THE ADDRESS ON THE STACK
043174  062716  000010                    ADD     #RMCS2,(SP)          ;FORM THE ADDRESS OF RMCS2
043200  032736  010000                    BIT     #BIT12,@(SP)+        ;CHECK THE 'NED' BIT
043204  001023                            BNE     WRT.R3               ;BR IF DRIVE NON-EXISTENT
043206  004037  042722                    JSR     R0,RD.RM             ;CHECK FOR PARITY ERROR ON WRITE
043212  000014                            RMER1
043214  043254                            WRT.R3
043216  032726  000010                    BIT     #BIT03,(SP)+
043222  001416                            BEQ     WRT.R4               ;BR IF 'PAR=0''
043224  016037  177776  043236            MOV     -2(R0),1$            ;PICKUP THE INDEX
043232  004037  042722                    JSR     R0,RD.RM             ;READ THE REG.
043236  000000                  1$:       .WORD   0                    ;REG. INDEX
043240  043254                            WRT.R3                       ;RETURN TO THIS ADDRESS ON ERROR
043242  104004                            EMT     4                    ;REPORT THE PARITY ON WRITE ERROR
043244  005726                            TST     (SP)+                ;CLEAR OFF THE STACK
043246  005327                            DEC     (PC)+                ;DECREMENT THE ERROR COUNT
043250  000003                  WRT.R2:   .WORD   3                    ;RETRY COUNTER
043252  002343                            BGE     WRT.R1               ;TRY AGAIN IF NOT FINISHED
043254  011000                  WRT.R3:   MOV     (R0),R0              ;TAKE THE 'PARITY ON WRITE'' ERROR EXIT
043256  000401                            BR      WRT.R5               ;EXIT
043260  005720                  WRT.R4:   TST     (R0)+                ;ADJUST FOR ERROR FREE EXIT
043262  000200                  WRT.R5:   RTS     R0

                                ;ROUTINE TO SAVE THE RH/RM REGISTERS AS PER DPB+14
                                ;
                                ;CALL
                                ;         MOV     #DPBNUM,R2           ;DPB POINTER TO R2
                                ;         JSR     PC,SVRH70            ;SAVE THE DRIVES REG'S

043264  104412                  SVRH70:   SAVREG                       ;SAVE R0 - R5
043266  005702                            TST     R2                   ;QUEUE ENTRY FOR THE DRIVE ?
043270  001442                            BEQ     6$                   ;BR IF NONE
043272  013704  035762                    MOV     RMADR,R4
043276  111264  000010                    MOVB    (R2),RMCS2(R4)       ;SELECT DRIVE
043302  016203  000014                    MOV     14(R2),R3            ;GET THE ERROR TABLE POINTER
043306  001433                            BEQ     6$                   ;EXIT IF NO ADDRESS
043310  005037  043344                    CLR     3$                   ;COUNTER & POINTER
043314  023727  043344  000022  1$:       CMP     3$,#RMDB             ;REACHED THE BUFFER REGISTER ?
043322  001006                            BNE     2$                   ;BR IF NOT
043324  032764  000200  000010            BIT     #BIT07,RMCS2(R4)             ;'OR' SET ?
043332  001002                            BNE     2$                   ;BR IF SET
043334  005023                            CLR     (R3)+                ;STORE RMDB AS ZEROES
043336  000405                            BR      4$                   ;CONTINUE
043340  004037  042722          2$:       JSR     R0,RD.RM             ;READ THE SELECTED REGISTER
043344  000000                  3$:       .WORD   0                    ;REGISTER INDEX
043346  043372                            5$                           ;ERROR RETURN ADDRESS
043350  012623                            MOV     (SP)+,(R3)+          ;STORE THE REGISTER CONTENTS
043352  023727  043344  000046  4$:       CMP     3$,#RMEC2            ;REACHED THE END ?
043360  001406                            BEQ     6$                   ;BR IF YES
```

```
043362  062737  000002  043344           ADD     #2,3$              ;INCREMENT THE REGISTER INDEX
043370  000751                           BR      1$                 ;CONTINUE READING THE REGISTERS
043372  004737  040122          5$:      JSR     PC,CI7             ;PROCESS THE UNCORRECTABLE PARITY ERROR
043376  104413                  6$:      RESREG                     ;RESTORE R0 - R5
043400  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

043402  010446                  SET.IE:  MOV     R4,-(SP)           ;SAVE R4
043404  013704  035762                   MOV     RMADR,R4           ;PICKUP ADDRESS OF RMCS1
043410  010164  000010                   MOV     R1,RMCS2(R4)       ;SELECT DRIVE
043414  011446                           MOV     (R4),-(SP)         ;READ RMCS1
043416  052716  040000                   BIS     #BIT14,(SP)        ;SET THE ''TRE'' BIT OF THE WORD READ
043422  000316                           SWAB    (SP)               ;ADJUST FOR DATO
043424  112714  000100                   MOVB    #BIT06,(R4)        ;SET ''IE''
043430  032764  010000  000010           BIT     #BIT12,RMCS2(R4)          ;IS 'NED'=1?
043436  001002                           BNE     1$                 ;YES--CLEAR ''TRE''
043440  005726                           TST     (SP)+              ;CLEAN OFF THE STACK
043442  000402                           BR      2$
043444  112664  000001          1$:      MOVB    (SP)+,1(R4)        ;CLEAR ''TRE''
043450  012604                  2$:      MOV     (SP)+,R4           ;RESTORE R4
043452  000207                           RTS     PC                 ;RETURN TO CALLER

                                ;QUEUE COUNT
043454     000                  QCNT:    .BYTE   0                  ;DRIVE 0
043455     000                           .BYTE   0                  ;DRIVE 1
043456     000                           .BYTE   0                  ;DRIVE 2
043457     000                           .BYTE   0                  ;DRIVE 3
043460     000                           .BYTE   0                  ;DRIVE 4
043461     000                           .BYTE   0                  ;DRIVE 5
043462     000                           .BYTE   0                  ;DRIVE 6
043463     000                           .BYTE   0                  ;DRIVE 7

                                ;QUEUE INPUT POINTERS

043464  043546                  QINPT:   .WORD   QDRV0              ;DRIVE 0
043466  043566                           .WORD   QDRV1              ;DRIVE 1
043470  043606                           .WORD   QDRV2              ;DRIVE 2
043472  043626                           .WORD   QDRV3              ;DRIVE 3
043474  043646                           .WORD   QDRV4              ;DRIVE 4
043476  043666                           .WORD   QDRV5              ;DRIVE 5
043500  043706                           .WORD   QDRV6              ;DRIVE 6
043502  043726                           .WORD   QDRV7              ;DRIVE 7

                                ;QUEUE OUTPUT POINTERS

043504  043546                  QOUTPT:  .WORD   QDRV0              ;DRIVE 0
043506  043566                           .WORD   QDRV1              ;DRIVE 1
043510  043606                           .WORD   QDRV2              ;DRIVE 2
043512  043626                           .WORD   QDRV3              ;DRIVE 3
043514  043646                           .WORD   QDRV4              ;DRIVE 4
043516  043666                           .WORD   QDRV5              ;DRIVE 5
043520  043706                           .WORD   QDRV6              ;DRIVE 6
```

```
043522  043726                          .WORD   QDRV7           ;DRIVE 7

043524  043546                  QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
043526  043566                  QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
043530  043606                          .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
043532  043626                          .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
043534  043646                          .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
043536  043666                          .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
043540  043706                          .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
043542  043726                          .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
043544  043746                          .WORD   QTERM           ;STOP DRIVE 7

                                ;DRIVE REQUEST QUEUES

043546                          QDRV0:  .BLKW   10
043566                          QDRV1:  .BLKW   10
043606                          QDRV2:  .BLKW   10
043626                          QDRV3:  .BLKW   10
043646                          QDRV4:  .BLKW   10
043666                          QDRV5:  .BLKW   10
043706                          QDRV6:  .BLKW   10
043726                          QDRV7:  .BLKW   10
        043746                  QTERM=.

                                ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
                                ;
                                ;CALL
                                ;       JSR     PC,CLRQUE

043746  104412                  CLRQUE: SAVREG                  ;SAVE R0 - R5
043750  012702  043454                  MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
043754  005022                          CLR     (R2)+           ;DRIVES 0 & 1
043756  005022                          CLR     (R2)+           ;DRIVES 2 & 3
043760  005022                          CLR     (R2)+           ;DRIVES 4 & 5
043762  005022                          CLR     (R2)+           ;DRIVES 6 & 7
043764  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING
043770  012701  043524                  MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
043774  012122                  1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
043776  005303                          DEC     R3
044000  001375                          BNE     1$
044002  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
044006  012701  043524                  MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
044012  012122                  2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
044014  005303                          DEC     R3
044016  001375                          BNE     2$
044020  104413                          RESREG                  ;RESTORE R0 - R5
044022  000207                          RTS     PC

                                ;EMPTY THE QUEUE SPECIFIED BY R1
                                ;
                                ;CALL
                                ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
                                ;       JSR     PC,EMPTYQ

044024  105061  043454          EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
044030  006301                          ASL     R1
044032  016161  043464  043504          MOV     QINPT(R1),QOUTPT(R1)    ;SET OUTPUT QUEUE POINTER=INPUT POINTER
```

```
044040 006201                          ASR     R1
044042 000207                          RTS     PC

                               ;ROUTINE TO PUT A REQUEST IN QUEUE
                               ;
                               ;CALL
                               ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER
                               ;       MOV     #DPB,R2           ;ADDRESS OF PARAMETER BLOCK
                               ;       JSR     R0,DRVQUE         ;GO PUT REQUEST IN QUEUE
                               ;       RETURN1                   ;RETURN HERE IF QUEUE IS FULL
                               ;       RETURN2                   ;RETURN HERE IF REQUEST IS IN QUEUE

044044 122761  000010  043454  DRVQUE: CMPB    #10,QCNT(R1)      ;IS QUEUE FULL?
044052 001421                          BEQ     2$                ;BR IF YES-TAKE RETURN1
044054 105261  043454                  INCB    QCNT(R1)          ;INCREMENT QUEUE COUNT
044060 006301                          ASL     R1
044062 010271  043464                  MOV     R2,@QINPT(R1)     ;PUT THIS REQUEST IN QUEUE
044066 062761  000002  043464          ADD     #2,QINPT(R1)      ;UPDATE THE QUEUE POINTER
044074 026161  043464  043526          CMP     QINPT(R1),QSTOP(R1)       ;TIME TO RESET THE POINTER
044102 001003                          BNE     1$                ;BR IF NO
044104 016161  043524  043464          MOV     QSTART(R1),QINPT(R1)      ;YES--RESET POINTER
044112 006201                  1$:     ASR     R1
044114 005720                          TST     (R0)+             ;TAKE RETURN 2
044116 000200                  2$:     RTS     R0                ;RETURN TO USER

                               ;ROUTINE TO GET THE ''DPB'' ADDRESS OF NEXT REQUEST IN QUEUE
                               ;
                               ;CALL
                               ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER TO R1
                               ;       JSR     PC,GETREQ         ;GO GET THE REQUEST
                               ;       RETURN                    ;R2=''DPB'' ADDRESS OF THE REQUEST
                               ;                                 ;R2=0 IF NO REQUEST IN QUEUE

044120 005002                  GETREQ: CLR     R2
044122 105761  043454                  TSTB    QCNT(R1)          ;IS THERE ANY REQUEST IN QUEUE?
044126 001404                          BEQ     2$                ;NO
044130 006301                  1$:     ASL     R1
044132 017102  043504                  MOV     @QOUTPT(R1),R2    ;PICKUP ''DPB'' POINTER FOR THIS DRIVE
044136 006201                          ASR     R1
044140 000207                  2$:     RTS     PC                ;RETURN TO USER

                               ;ROUTINE TO ''POP'' THE REQUEST FROM QUEUE
                               ;
                               ;CALL
                               ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER TO R1
                               ;       JSR     PC,POPQUE         ;CALL TO REMOVE REQUEST
                               ;       RETURN                    ;R2=ADDRESS OF DPB REMOVED

044142 105361  043454          POPQUE: DECB    QCNT(R1)          ;DECREMENT QUEUE COUNT
044146 006301                          ASL     R1
044150 017102  043504                  MOV     @QOUTPT(R1),R2    ;GET THE ''DPB'' POINTER
044154 005071  043504                  CLR     @QOUTPT(R1)       ;REMOVE DPB ADDRESS FROM THE QUEUE
044160 062761  000002  043504          ADD     #2,QOUTPT(R1)     ;UPDATE THE QUEUE POINTER
044166 026161  043504  043526          CMP     QOUTPT(R1),QSTOP(R1)      ;TIME TO RESET THE POINTER?
044174 001003                          BNE     1$                ;NO--BR TO EXIT
044176 016161  043524  043504          MOV     QSTART(R1),QOUTPT(R1)     ;YES--RESET THE POINTER
044204 006201                  1$:     ASR     R1
```

```
044206  000207                  RTS     PC                      ;RETURN TO USER



                        .SBTTL  DATA, CONTROL, & STATUS BLOCKS

                        ;BLOCK LOCATION EQUATE STATEMENTS

        000001          $FMT    =       1                       ;FMT,HCI,ECI OR OFFSET CODE
        000002          $COMND  =       $FMT+1                  ;OPERATION CODE
        000003          $PSEL   =       $FMT+2                  ;PORT SELECT & BITS A16, A17
        000004          $WRDM   =       $FMT+3                  ;WORD COUNT (2'S COMP)
        000006          $BUF    =       $FMT+5                  ;BUFFER ADDR OR REGISTER TABLE POINTER
        000010          $SEC    =       $FMT+7                  ;SECTOR ADDRESS OR 1ST REG ADDR
        000011          $TRK    =       $FMT+10                 ;TRACK ADDRESS OF LAST REG ADDR
        000012          $CYL    =       $FMT+11                 ;CYLINDER ADDR
        000014          $REG    =       $FMT+13                 ;REGISTER STORAGE (IF ERROR)
        000016          $STATUS =       $FMT+15                 ;STATUS WORD (SET BY DRIVER)

                        ;DRIVE'S HISTORY AND CURRENT INDICATOR STORAGE EQUATES

        000020          $WRDL   =       $FMT+17                 ;WORD COUNT (NOT 2'S COMP)
        000022          $SSEC   =       $WRDL+2                 ;SECTOR SIZE FOR CURRENT OPERATION
        000024          $CODE   =       $WRDL+4                 ;PRESENT COMMAND SELECTION CODE
        000026          $PACK   =       $WRDL+6                 ;WRITE DATA PACK INDICATOR
        000027          $PREVO  =       $WRDL+7                 ;PREVIOUS COMMAND SELECTION CODE
        000030          $PATTC  =       $WRDL+10                ;PATTERN CODE
        000032          $PREVA  =       $WRDL+12                ;PREVIOUS ADDRESS - TRACK, SECTOR, CYLINDER
        000036          $OPERC  =       $WRDL+16                ;OPERATION COUNT
        000042          $POSIT  =       $WRDL+22                ;SEEK COUNT
        000046          $TRANS  =       $WRDL+26                ;TOTAL BITS XFERED COUNT (R & W)
        000052          $READ   =       $WRDL+32                ;TOTAL BITS READ COUNT
        000056          $TOTAL  =       $WRDL+36                ;TOTAL ERRORS (ALL TYPES) COUNT
        000060          $SOFT   =       $WRDL+40                ;'SOFT' ERROR COUNT
        000062          $HARD   =       $WRDL+42                ;'HARD' ERROR COUNT
        000064          $SKI    =       $WRDL+44                ;'SKI' OR 'OCYL' ERROR COUNT
        000066          $MISPO  =       $WRDL+46                ;PROG DETECTED MISPOSITIONING ERROR S COUNT
        000070          $PASSC  =       $WRDL+50                ;PASS COUNTER
        000072          $FAIR   =       $WRDL+52                ;OPERATION QUEUE 'FAIRNESS' COUNT

                        ;INDEX EQUATES TO THE NEXT OPERATION PARAMETERS

        000074          $NCODE  =       $WRDL+54                ;NEXT OPERATION CODE
        000075          $NPATC  =       $NCODE+1                ;NEXT PATTERN
        000076          $NSEC   =       $NCODE+2                ;NEXT SECTOR
        000077          $NTRK   =       $NCODE+3                ;NEXT TRACK
        000100          $NCYL   =       $NCODE+4                ;NEXT CYLINDER
        000102          $NWRDL  =       $NCODE+6                ;NEXT BUFFER SIZE
        000104          $NEXT   =       $NCODE+10               ;PARAMETER SELECTION INDICATOR

                        ;INDEX EQUATES FOR MAXIMUM/MINIMUM ADDRESSES

        000106          MAXCYL  =       $NCODE+12               ;MAXIMUM CYLINDER ADDRESS
        000110          MINCYL  =       MAXCYL+2                ;MINIMUM CYLINDER ADDRESS
        000112          MAXTRK  =       MAXCYL+4                ;MAXIMUM TRACK ADDRESS
        000114          MINTRK  =       MAXCYL+6                ;MINIMUM TRACK ADDRESS
        000116          MAXSEC  =       MAXCYL+10               ;MAXIMUM SECTOR ADDRESS
        000120          MINSEC  =       MAXCYL+12               ;MINIMUM SECTOR ADDRESS
```

```
  1471           000122                         $FIRST  =       MAXCYL+14       ;FIRST OPERATION INDICATOR
  1472
  1473                                          ;BAD SECTOR/TRACK ADDRESS STORAGE AREA INDEX EQUATE
  1474
  1475           000124                         $BDSEC  =       MAXCYL+16       ;BAD SECTOR STORAGE TABLE PLUS TERMINATOR
  1476
  1477                                          ;DRIVE ID AREA INDEX EQUATE
  1478
  1479           002106                         $DRVID  =       $BDSEC+<126.*8.>+2      ;DRIVE ID
  1480
  1481                                          ;RH/RM REGISTER EQUATES
  1482
  1483           002116                         $RMCS1  =       $DRVID+10       ;RM REGISTER STORAGE
  1484           002120                         $RMWC   =       $RMCS1+2
  1485           002122                         $RMBA   =       $RMCS1+4
  1486           002124                         $RMDA   =       $RMCS1+6
  1487           002126                         $RMCS2  =       $RMCS1+10
  1488           002130                         $RMDS   =       $RMCS1+12
  1489           002132                         $RMER1  =       $RMCS1+14
  1490           002134                         $RMAS   =       $RMCS1+16
  1491           002136                         $RMLA   =       $RMCS1+20
  1492           002140                         $RMDB   =       $RMCS1+22
  1493           002142                         $RMMR1  =       $RMCS1+24
  1494           002144                         $RMDT   =       $RMCS1+26
  1495           002146                         $RMSN   =       $RMCS1+30
  1496           002150                         $RMOF   =       $RMCS1+32
  1497           002152                         $RMDC   =       $RMCS1+34
  1498           002154                         $RMHR   =       $RMCS1+36
  1499           002156                         $RMMR2  =       $RMCS1+40
  1500           002160                         $RMER2  =       $RMCS1+42
  1501           002162                         $RMEC1  =       $RMCS1+44
  1502           002164                         $RMEC2  =       $RMCS1+46
  1503
  1511                                          ;BLOCK FOR DRIVE 0
       044210       000      000                DRIVE0: .BYTE   0,0             ;DRIVE NUMBER 0
                                                        .BLKW   5
       044224   046326                                  .WORD   .+$RMCS1-$REG
                                                        .BLKB   $RMEC2-$REG

                                                ;BLOCK FOR DRIVE 1
       046376       001      000                DRIVE1: .BYTE   1,0             ;DRIVE NUMBER 1
                                                        .BLKW   5
       046412   050514                                  .WORD   .+$RMCS1-$REG
                                                        .BLKB   $RMEC2-$REG

                                                ;BLOCK FOR DRIVE 2
       050564       002      000                DRIVE2: .BYTE   2,0             ;DRIVE NUMBER 2
                                                        .BLKW   5
       050600   052702                                  .WORD   .+$RMCS1-$REG
                                                        .BLKB   $RMEC2-$REG

                                                ;BLOCK FOR DRIVE 3
       052752       003      000                DRIVE3: .BYTE   3,0             ;DRIVE NUMBER 3
                                                        .BLKW   5
       052766   055070                                  .WORD   .+$RMCS1-$REG
                                                        .BLKB   $RMEC2-$REG
```

```
                                            ;BLOCK FOR DRIVE 4
     055140       004      000              DRIVE4: .BYTE   4,0                     ;DRIVE NUMBER 4
                                                    .BLKW   5
     055154  057256                                 .WORD   .+$RMCS1-$REG
                                                    .BLKB   $RMEC2-$REG

                                            ;BLOCK FOR DRIVE 5
     057326       005      000              DRIVE5: .BYTE   5,0                     ;DRIVE NUMBER 5
                                                    .BLKW   5
     057342  061444                                 .WORD   .+$RMCS1-$REG
                                                    .BLKB   $RMEC2-$REG

                                            ;BLOCK FOR DRIVE 6
     061514       006      000              DRIVE6: .BYTE   6,0                     ;DRIVE NUMBER 6
                                                    .BLKW   5
     061530  063632                                 .WORD   .+$RMCS1-$REG
                                                    .BLKB   $RMEC2-$REG

                                            ;BLOCK FOR DRIVE 7
     063702       007      000              DRIVE7: .BYTE   7,0                     ;DRIVE NUMBER 7
                                                    .BLKW   5
     063716  066020                                 .WORD   .+$RMCS1-$REG
                                                    .BLKB   $RMEC2-$REG


                                            ;GENERAL PURPOSE PARAMETER BLOCK

     066070       000                       GENDPB: .BYTE   0                       ;DRIVER PARAMETER BLOCK, DRIVE #
     066071       000                               .BYTE   0                       ;OFFSET VALUE OR FMT16, HCI OR ECI
     066072       000                               .BYTE   0                       ;COMMAND CODE
     066073       000                               .BYTE   0                       ;PSEL, A16 AND A17
     066074  177776                                 .WORD   -2                      ;WORD COUNT (NEG)
     066076  076746                                 .WORD   CYLNDR                  ;BUFFER ADDRESS
     066100       000                               .BYTE   0                       ;SECTOR ADDRESS
     066101       000                               .BYTE   0                       ;TRACK ADDRESS
     066102  000000                                 .WORD   0                       ;CYLINDER ADDRESS
     066104  066110                                 .WORD   GENREG                  ;ADDRESS TO SAVE ALL RH/RM REG'S
     066106  000000                                 .WORD   0                       ;STATUS WORD

     066110                                 GENREG: .BLKW   24                      ;REGISTER STORAGE

```

```
                                            .SBTTL  ERROR MESSAGES

  066160     122      110      061     EM1:       .ASCIZ  @RH11/RH70 INTERRUPT OCCURRED (RMAS = 0)@
  066230     125      116      105     EM2:       .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
  066266     115      101      123     EM3:       .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
  066324     115      101      123     EM4:       .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
  066361     101      104      104     EM5:       .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
  066415     122      110      061     EM6:       .ASCIZ  @RH11/RH70 DIDN'T RESPOND TO ADDRESSING@
  066464     125      116      103     EM10:      .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
  066527     106      101      124     EM11:      .ASCIZ  /FATAL MASSBUS PARITY ERROR/
  066562     120      105      122     EM12:      .ASCIZ  /PERSISTENT DEVICE UNSAFE/
  066613     117      120      105     EM13:      .ASCIZ  /OPERATION NOT COMPLETED WITHIN TIME LIMIT/
  066665     104      122      111     EM14:      .ASCIZ  /DRIVE WENT OFFLINE/
  066710     116      117      040     EM15:      .ASCIZ  /NO RESPONSE TO PORT REQUEST/
  066744     110      105      101     EM20:      .ASCIZ  /HEADER CRC ERROR/
  066765     104      101      124     EM21:      .ASCIZ  /DATA CHECK ('DCK') ERROR/
  067016     127      122      111     EM22:      .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') SET/
  067071     127      122      111     EM23:      .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') NOT SET/
  067150     110      105      101     EM24:      .ASCIZ  /HEADER READ ERROR - 'FMT' BIT DROPPED/
  067216     110      105      101     EM25:      .ASCIZ  /HEADER READ ERROR - HEADER COMPARE ('HCE') ERROR/
  067277     106      117      122     EM26:      .ASCIZ  /FORMAT ERROR ('FER')/
  067324     110      105      101     EM27:      .ASCIZ  /HEADER COMPARE ('HCE') ERROR/
  067361     115      111      123     EM30:      .ASCIZ  /MISCELLANEOUS DRIVE ERROR/
  067413     117      120      105     EM31:      .ASCIZ  /OPERATION INCOMPLETE ('OPI') ERROR/
  067456     104      122      111     EM32:      .ASCIZ  /DRIVE TIMING ('DTE') ERROR/
  067511     120      101      122     EM33:      .ASCIZ  /PARITY ('PAR') ERROR AFTER OPERATION STARTED/
  067566     127      122      111     EM34:      .ASCIZ  /WRITE CLOCK FAILURE ('WCF') ERROR/
  067630     111      116      126     EM35:      .ASCIZ  /INVALID ADDRESS ('IAE') ERROR/
  067666     127      122      111     EM36:      .ASCIZ  /WRITE LOCK ('WLE') ERROR/
  067717     104      101      124     EM37:      .ASCIZ  /DATA CHECK ('DCK') SET DURING WRITE CHECK COMMAND/
  070001     122      110      061     EM40:      .ASCIZ  @RH11/RH70 OR UNIBUS TRANSFER ERROR@
  070044     102      125      123     EM41:      .ASCIZ  /BUS ADDRESS OR WORD COUNT INCORRECT/
  070110     104      101      124     EM42:      .ASCIZ  /DATA COMPARE ERRORS - NO OTHER ERROR(S) DETECTED/
  070171     103      101      116     EM43:      .ASCIZ  /CAN'T MATCH DATA READ WITH A PATTERN/
  070236     105      122      122     EM44:      .ASCIZ  @ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11/RH70@
  070327     105      103      103     EM45:      .ASCIZ  /ECC LOGIC FAILURE - POSITION REGISTER VALUE NOT VALID/
  070415     102      125      123     EM46:      .ASCIZ  /BUS ADDRESS AND WORD COUNT NOT CONSISTENT/
  070467     123      105      105     EM50:      .ASCIZ  /SEEK INCOMPLETE ('SKI') ERROR/
  070525     120      122      117     EM51:      .ASCIZ  /PROGRAM DETECTED POSITIONING ERROR/
  070570     104      122      111     EM60:      .ASCIZ  /DRIVE UNSAFE ERROR/

```

```
070613     122     115     101  DH1:     .ASCIZ  /RMAS/
070620     104     122     111  DH2:     .ASCIZ  /DRIVE   RMDS    RMER1   RMER2  RMMR2   RMAS/
070674     104     122     111  DH3:     .ASCIZ  /DRIVE   REG ADR  DATA/
070722     104     122     111  DH4:     .ASCIZ  /DRIVE   REG ADR    GOOD    BAD/
070761     044     122     115  DH6:     .ASCIZ  /$RMADR/
070770     104     122     126  DH14:    .ASCII  /DRV RMCS1   RMCS2   RMDS    RMER1   /
071034     122     115     115           .ASCIZ  /RMMR2  RMER2   RMEC1   RMEC2/<CR><LF>
071073     122     115     127  DH15:    .ASCII  /RMWC    RMBA    RMDA    RMAS    RMLA    /
071143     122     115     104           .ASCIZ  /RMDB    RMMR1   RMDT/<CR><LF>
071172     122     115     123  DH16:    .ASCIZ  /RMSN    RMOF    RMDC    RMHR    STATUS/<CR><LF>

                                .EVEN
```

```
 071244  001344  000000          DT1:     .WORD   ATTN,0
 071250  001220  035614  035616  DT2:     .WORD   DRIVE,RMERRS,RMERRS+2,RMERRS+4,RMERRS+6,ATTN,0
 071266  001220  042746  042750  DT3:     .WORD   DRIVE,RD.ADR,RD.WRD,0
 071276  001220  043166  043164  DT4:     .WORD   DRIVE,WRT.AD,WRT.WD,RD.WRD,0
 071310  001272  000000          DT6:     .WORD   $RMADR,0
 071314  002116  002126  002130  DT14:    .WORD   $RMCS1,$RMCS2,$RMDS,$RMER1,$RMMR2,$RMER2,$RMEC1,$RMEC2,0
 071336  002120  002122  002124  DT15:    .WORD   $RMWC,$RMBA,$RMDA,$RMAS,$RMLA,$RMDB,$RMMR1,$RMDT,0
 071360  002146  002150  002152  DT16:    .WORD   $RMSN,$RMOF,$RMDC,$RMHR,$TATUS,0
```

```
  071374     120     122     105  LIN2C:   .ASCIZ  /PRESENT ORDER = /
  071415     040     040     120  LIN2P:   .ASCIZ  /  PREVIOUS ORDER = /
  071441     052     040     105  LIN2S:   .ASCIZ  @* ERROR AT BAD TRACK/SECTOR@
  071475     105     122     122  LINM3:   .ASCIZ  /ERROR AT C/
  071510     040     124     000  T:       .ASCIZ  / T/
  071513     120     122     105  LINN3:   .ASCIZ  /PRESENT ADDR = C/
  071534     040     123     000  S:       .ASCIZ  / S/
  071537     040     040     040  LINP3:   .ASCIZ  /    PREV ADDR = C/
  071560     123     124     101  LINS3:   .ASCIZ  /START CYL = /
  071575     040     040     105  LINEN3:  .ASCIZ  /  END CYL = /
  071612     040     040     101  LINA3:   .ASCIZ  /  ACTUAL CYL = /
  071632     040     040     124  LINT3:   .ASCIZ  /  TRK = /
  071643     040     122     115  LINCA3:  .ASCIZ  / RMDC = /
  071654     122     115     104  LINDA3:  .ASCIZ  /RMDA = /
  071664     122     115     102  LINB3:   .ASCIZ  /RMBA = /
  071674     040     040     122  LINW3:   .ASCIZ  /  RMWC = /
  071706     123     124     101  LINST3:  .ASCIZ  /START TRK = /
  071723     123     124     101  LINSS3:  .ASCIZ  /START SEC = /
  071740     102     125     106  LINM4:   .ASCIZ  /BUFFER ADDR = /
  071757     040     040     123  LINS4:   .ASCIZ  /  SIZE = /
  071771     040     040     101  LINX4:   .ASCIZ  /  ACTUAL NMBR WRDS XFRD = /
  072024     107     117     117  LIND5:   .ASCIZ  /GOOD DATA = /
  072041     040     040     102  LINB5:   .ASCIZ  /  BAD DATA = /
  072057     040     040     127  LINP5:   .ASCIZ  /  WORD POS = /
  072075     110     105     101  LINS5:   .ASCIZ  /HEADER FROM ERROR SECTOR = /
  072131     122     115     105  LINEP5:  .ASCIZ  /RMEC1 = /
  072142     040     122     115  LINEO5:  .ASCIZ  / RMEC2 = /
  072154     123     105     103  LINB6:   .ASCIZ  /SECTOR IS ECC CORRECTABLE /
  072207     123     105     103  LINC6:   .ASCIZ  /SECTOR READ CORRECTLY /
  072236     103     117     122  LING6:   .ASCIZ  /CORRECTED ON /
  072254     040     122     105  LINR6:   .ASCIZ  / RETRIES/
  072265     125     116     103  LINU06:  .ASCIZ  /UNCORRECTABLE AFTER /
  072312     040     040     124  LIN7M:   .ASCIZ  /  TOTAL MISPOS ERR = /
  072340     117     122     104  LIN7O:   .ASCIZ  /ORDERS:/
  072350     040     124     117  LIN7P:   .ASCIZ  / TOTAL SEEKS = /
  072370     040     124     117  LIN7S:   .ASCIZ  / TOTAL SKI ERR = /
  072412     040     040     105  LIN7T:   .ASCIZ  /  ERRORS:/
  072424     040     040     127  LIN7X:   .ASCIZ  /  WRDS XFR:/
  072440     040     040     127  LIN7R:   .ASCIZ  /  WRDS READ:/
  072455     105     122     122  LIN8M:   .ASCIZ  /ERROR DURING RETRY/
  072500     104     101     124  LIN9B:   .ASCIZ  /DATA COMPARISON ERRORS/
  072527     040     040     040  LIN9H:   .ASCII  /           GOOD     BAD/<CR><LF>
  072555     114     117     103           .ASCIZ  /LOC        DATA     DATA/<CR><LF>
  072605     114     117     103  LIN9I:   .ASCIZ  /LOC        DATA/<CR><LF>
  072625     124     117     124  LIN9E:   .ASCIZ  /TOTAL COMPARE ERRORS = /
  072655     124     110     105  LIN9G:   .ASCIZ  /THE DATA COMPARED OK/<CR><LF>
  072704     105     122     122  LIN10A:  .ASCIZ  /ERROR BURST BEGINS AT WORD /
  072740     040     111     116  LIN10B:  .ASCIZ  / IN DATA FIELD OF ERROR SECTOR/<CR><LF>
  073001     105     122     122  LIN10C:  .ASCII  /ERROR WAS NOT IN THE DATA READ - /<CR><LF>
  073044     105     103     103           .ASCIZ  /ECC CORRECTION CAN'T BE PERFORMED/
  073106     105     103     103  LIN10H:  .ASCII  /ECC CORRECTION RESULTS/<CR><LF>
  073136     101     104     104           .ASCIZ  /ADDR    BAD       CORRECTED /<CR><LF>
  073173     103     117     116  LIN11H:  .ASCIZ  /CONTENTS OF ERROR SECTOR (REPORTED ABOVE)/<CR><LF>
  073247     101     104     104           .ASCIZ  /ADDR     DATA/<CR><LF>
  073266     040                  BLNKS4:  .ASCII  / /
  073267     040                  BLNKS3:  .ASCII  / /
  073270     040                  BLNKS2:  .ASCII  / /
```

```
     58 073271      040     000              BLNKS1: .ASCIZ  / /
     59 073273      122     105      124     LINX5:  .ASCIZ  /RETRIEVED BY A RDHD COMMAND/
     60
```

```
                                        .SBTTL  TELETYPE MESSAGES

  073327      077     000               QUES:    .ASCIZ  /?/
  073331      104     122        111    UNTMSG:  .ASCIZ  /DRIVE/
  073337      040     117        106    UNTOFF:  .ASCIZ  / OFFLINE/
  073350      040     117        116    UNTON:   .ASCIZ  / ONLINE/
  073360      040     116        117    UNTNOT:  .ASCIZ  / NOT BEING TESTED/
  073402      040     101        114    UNTASN:  .ASCIZ  / ALREADY BEING TESTED/
  073430      040     116        117    NOTRM:   .ASCIZ  @ NOT AN RM05/3/2@
  073451      040     116        117    NOTPRS:  .ASCIZ  / NOT PRESENT/
  073466      040     116        117    NOTAVL:  .ASCIZ  / NOT AVAILABLE/
  073505      040     125        116    NOTSAF:  .ASCIZ  / UNSAFE/
  073515      040     111        123    LODEV:   .ASCIZ  / IS LOAD DEVICE/
  073535      200     125        116    SYSTAT:  .ASCIZ  <CRLF>/UNIT STATUS:/
  073553      122     115        060    $RM02:   .ASCIZ  /RM02/
  073560      122     115        060    $RM03:   .ASCIZ  /RM03/
  073565      122     115        060    $RM05:   .ASCIZ  /RM05/
  073572      200     012        052    REPHD:   .ASCIZ  <CRLF><LF>/* * * * * PERFORMANCE REPORT * * * * * /
  073644      104     122        111    STATHD:  .ASCII  /DRIVE PERFORMANCE SUMMARY/<CRLF>
  073676      104     122        126             .ASCII  /DRV PASS ORDERS   SEEKS   WRDS XFER   WRDS READ /
  073756      123     117        106             .ASCIZ  /SOFT HARD  SKI MISP OTHER/<CRLF>
  074011      007     077        106    DROPNG:  .ASCIZ  <07>/?FATAL OR EXCESSIVE ERRORS/
  074045      200     105        116    ENDPAS:  .ASCIZ  <CRLF>/END OF PASS #/
  074064      040     117        116    MSGON:   .ASCIZ  / ON /
  074071      200     105        116    ENDTST:  .ASCIZ  <CRLF>/END OF TEST /
  074107      106     117        122    MSGFOR:  .ASCIZ  /FOR /
  074114      200     104        122    DEASSG:  .ASCIZ  <CRLF>/DRIVE DEASSIGNED/
  074136      200     052        052    DRNUM:   .ASCIZ  <CRLF>/********** DRIVE #/
  074162      040     123        124    ASGND:   .ASCIZ  / STARTED/
  074173      200     007        077    NEDCLK:  .ASCIZ  <CRLF><07>/? 'L' OR 'P' CLOCK REQUIRED ON SYSTEM/<CRLF>
  074244      116     000               N:       .ASCIZ  /N/
  074246      131     000               Y:       .ASCIZ  /Y/
  074250      056     000               PERIOD:  .ASCIZ  /./
  074252      040     077        103    MSWRO:   .ASCIZ  / ?CAN'T WRITE WITH SW00 SET/<CRLF>
  074307      040     077        111    INVLD:   .ASCIZ  / ?INVALID COMMAND/<CRLF>
  074332      200     105        116    ENTCOM:  .ASCIZ  <CRLF>/ENTER COMMAND: /
  074353      200     105        116    ENTDRV:  .ASCIZ  <CRLF>/ENTER DRIVE I.D.: /
  074377      200     105        116    ENTLMT:  .ASCIZ  <CRLF>/ENTER ADDRESS LIMITS:/<CRLF>
  074427      200     105        116    ENTADR:  .ASCIZ  <CRLF>@ENTER BAD SPOT ADDRESSES: @<CRLF>
  074464      072     000               COLON:   .ASCIZ  /:/
  074466      200     104        101    DATEIS:  .ASCIZ  <CRLF>/DATE: /
  074476      200     117        120    IDIS:    .ASCIZ  <CRLF>/OPERATOR I.D.: /
  074517      200     012        104    HEDLIN:  .ASCIZ  <CRLF><LF>/DRV  DRV I.D./<CRLF>
  074540      116     117        116    NONE:    .ASCIZ  /NONE/<CRLF>
  074546      040     077        111    BADENT:  .ASCIZ  / ?INVALID ENTRY/<CRLF>
  074567      007     123        131    BUSY:    .ASCIZ  <07>/SYSTEM BUSY.../<CRLF>
  074610      200     120        122    INTDON:  .ASCII  <CRLF>/PROGRAM INITIALIZATION COMPLETE/<CRLF>
  074651      124     131        120             .ASCIZ  /TYPE A 'CONTROL C' TO ENTER COMMANDS/
  074716      200     106        101    MERR1:   .ASCIZ  <CRLF>/FAILED TO RETRIEVE BAD SPOT FILE(DEC144) ON /
  074774      200     111        116    MERR2:   .ASCIZ  <CRLF>/INVALID FILE(DEC144) STRUCTURE ON /
  075040      040     077        102    MSFULL:  .ASCIZ  / ?BAD SPOT TABLE IS FULL/<CRLF>
  075072      103     131        114    MSGCYL:  .ASCIZ  /CYLNDR/
  075101      124     122        101    MSGTRK:  .ASCIZ  /TRACK /
  075110      123     105        103    MSGSEC:  .ASCIZ  /SECTOR/

                                        .EVEN

```

```
                                                ;PARAMETER ENTRY TABLE

 075120  075250  020000  001466         PARLST: .WORD   PAR1,8192.,MAXDL
 075126  075260  177777  001472                 .WORD   PAR2,-1,INTRVL
 075134  075430  177777  001464                 .WORD   PAR19,-1,PASCNT
 075142  075270  000017  001514                 .WORD   PAR3,15.,PATTEN
 075150  075370  000001  001502                 .WORD   PAR11,1,WCSEL
 075156  075400  000007  001504                 .WORD   PAR14,7,RATIO
 075164  075420  000001  001512                 .WORD   PAR16,1,ENDET
 075172  075360  000001  001500                 .WORD   PAR10,1,FORMAT
                                        ;       .WORD   PAR15,1,AUTOCK
 075200  075440  000001  001510                 .WORD   PAR20,1,NOTPRT
 075206  075450  000001  001516                 .WORD   PAR21,1,HEADER  ;RANDOM ADDRESS FLAG
 075214  000000                                 .WORD   0               ;TABLE TERMINATOR

 075216     040     057     040         SLASH:  .ASCIZ  @ / @
 075222     200     103     110         ASKPAR: .ASCIZ  <CRLF>/CHANGE PARAMETERS ? /
 075250     123     111     132         PAR1:   .ASCIZ  /SIZE    /
 075260     115     111     116         PAR2:   .ASCIZ  /MINUTE /
 075270     120     101     124         PAR3:   .ASCIZ  /PATTERN/
 075300     115     101     130         PAR4:   .ASCIZ  /MAXCYL /
 075310     115     111     116         PAR5:   .ASCIZ  /MINCYL /
 075320     115     101     130         PAR6:   .ASCIZ  /MAXTRK /
 075330     115     111     116         PAR7:   .ASCIZ  /MINTRK /
 075340     115     101     130         PAR8:   .ASCIZ  /MAXSEC /
 075350     115     111     116         PAR9:   .ASCIZ  /MINSEC /
 075360     106     117     122         PAR10:  .ASCIZ  /FORMAT /
 075370     122     101     116         PAR11:  .ASCIZ  /RANDOM /
 075400     122     101     124         PAR14:  .ASCIZ  /RATIO  /
 075410     103     110     105         PAR15:  .ASCIZ  /CHECK  /
 075420     105     116     104         PAR16:  .ASCIZ  /ENDING /
 075430     120     101     123         PAR19:  .ASCIZ  /PASSES /
 075440     115     105     123         PAR20:  .ASCIZ  /MESSAGE/
 075450     110     105     101         PAR21:  .ASCIZ  /HEADER /

                                        .EVEN


                                        ;PARAMETER TABLE POINTERS FOR ADDRESS LIMITS
 075460  075500                         TABLE:  .WORD   TABLE0          ;PARAMETER TABLE FOR DRIVE 0
 075462  075546                                 .WORD   TABLE1          ;PARAMETER TABLE FOR DRIVE 1
 075464  075614                                 .WORD   TABLE2          ;PARAMETER TABLE FOR DRIVE 2
 075466  075662                                 .WORD   TABLE3          ;PARAMETER TABLE FOR DRIVE 3
 075470  075730                                 .WORD   TABLE4          ;PARAMETER TABLE FOR DRIVE 4
 075472  075776                                 .WORD   TABLE5          ;PARAMETER TABLE FOR DRIVE 5
 075474  076044                                 .WORD   TABLE6          ;PARAMETER TABLE FOR DRIVE 6
 075476  076112                                 .WORD   TABLE7          ;PARAMETER TABLE FOR DRIVE 7


                                        ;PARAMETER TABLE FOR ADDRESS LIMITS
 075500  075310  001465  044320         TABLE0: .WORD   PAR5,821.,MINCYL+DRIVE0
 075506  075300  001465  044316                 .WORD   PAR4,821.,MAXCYL+DRIVE0
 075514  075330  000000  044324                 .WORD   PAR7,0,MINTRK+DRIVE0
 075522  075320  000000  044322                 .WORD   PAR6,0,MAXTRK+DRIVE0
 075530  075350  000037  044330                 .WORD   PAR9,31.,MINSEC+DRIVE0
 075536  075340  000037  044326                 .WORD   PAR8,31.,MAXSEC+DRIVE0
 075544  000000                                 .WORD   0               ;TERMINATOR

 075546  075310  001465  046506         TABLE1: .WORD   PAR5,821.,MINCYL+DRIVE1
```

```
        075554  075300  001465  046504                  .WORD   PAR4,821.,MAXCYL+DRIVE1
        075562  075330  000000  046512                  .WORD   PAR7,0,MINTRK+DRIVE1
        075570  075320  000000  046510                  .WORD   PAR6,0,MAXTRK+DRIVE1
        075576  075350  000037  046516                  .WORD   PAR9,31.,MINSEC+DRIVE1
        075604  075340  000037  046514                  .WORD   PAR8,31.,MAXSEC+DRIVE1
        075612  000000                                  .WORD   0               ;TERMINATOR

        075614  075310  001465  050674  TABLE2: .WORD   PAR5,821.,MINCYL+DRIVE2
        075622  075300  001465  050672                  .WORD   PAR4,821.,MAXCYL+DRIVE2
        075630  075330  000000  050700                  .WORD   PAR7,0,MINTRK+DRIVE2
        075636  075320  000000  050676                  .WORD   PAR6,0,MAXTRK+DRIVE2
        075644  075350  000037  050704                  .WORD   PAR9,31.,MINSEC+DRIVE2
        075652  075340  000037  050702                  .WORD   PAR8,31.,MAXSEC+DRIVE2
        075660  000000                                  .WORD   0               ;TERMINATOR

        075662  075310  001465  053062  TABLE3: .WORD   PAR5,821.,MINCYL+DRIVE3
        075670  075300  001465  053060                  .WORD   PAR4,821.,MAXCYL+DRIVE3
        075676  075330  000000  053066                  .WORD   PAR7,0,MINTRK+DRIVE3
        075704  075320  000000  053064                  .WORD   PAR6,0,MAXTRK+DRIVE3
        075712  075350  000037  053072                  .WORD   PAR9,31.,MINSEC+DRIVE3
        075720  075340  000037  053070                  .WORD   PAR8,31.,MAXSEC+DRIVE3
        075726  000000                                  .WORD   0               ;TERMINATOR

        075730  075310  001465  055250  TABLE4: .WORD   PAR5,821.,MINCYL+DRIVE4
        075736  075300  001465  055246                  .WORD   PAR4,821.,MAXCYL+DRIVE4
        075744  075330  000000  055254                  .WORD   PAR7,0,MINTRK+DRIVE4
        075752  075320  000000  055252                  .WORD   PAR6,0,MAXTRK+DRIVE4
        075760  075350  000037  055260                  .WORD   PAR9,31.,MINSEC+DRIVE4
        075766  075340  000037  055256                  .WORD   PAR8,31.,MAXSEC+DRIVE4
        075774  000000                                  .WORD   0               ;TERMINATOR

        075776  075310  001465  057436  TABLE5: .WORD   PAR5,821.,MINCYL+DRIVE5
        076004  075300  001465  057434                  .WORD   PAR4,821.,MAXCYL+DRIVE5
        076012  075330  000000  057442                  .WORD   PAR7,0,MINTRK+DRIVE5
        076020  075320  000000  057440                  .WORD   PAR6,0,MAXTRK+DRIVE5
        076026  075350  000037  057446                  .WORD   PAR9,31.,MINSEC+DRIVE5
        076034  075340  000037  057444                  .WORD   PAR8,31.,MAXSEC+DRIVE5
        076042  000000                                  .WORD   0               ;TERMINATOR

        076044  075310  001465  061624  TABLE6: .WORD   PAR5,821.,MINCYL+DRIVE6
        076052  075300  001465  061622                  .WORD   PAR4,821.,MAXCYL+DRIVE6
        076060  075330  000000  061630                  .WORD   PAR7,0,MINTRK+DRIVE6
        076066  075320  000000  061626                  .WORD   PAR6,0,MAXTRK+DRIVE6
        076074  075350  000037  061634                  .WORD   PAR9,31.,MINSEC+DRIVE6
        076102  075340  000037  061632                  .WORD   PAR8,31.,MAXSEC+DRIVE6
        076110  000000                                  .WORD   0               ;TERMINATOR

        076112  075310  001465  064012  TABLE7: .WORD   PAR5,821.,MINCYL+DRIVE7
        076120  075300  001465  064010                  .WORD   PAR4,821.,MAXCYL+DRIVE7
        076126  075330  000000  064016                  .WORD   PAR7,0,MINTRK+DRIVE7
        076134  075320  000000  064014                  .WORD   PAR6,0,MAXTRK+DRIVE7
        076142  075350  000037  064022                  .WORD   PAR9,31.,MINSEC+DRIVE7
        076150  075340  000037  064020                  .WORD   PAR8,31.,MAXSEC+DRIVE7
        076156  000000                                  .WORD   0               ;TERMINATOR


                                        ;ROUTINE TO GET THE DATE AND THE OPERATOR FROM THE OPERATOR
```

```
                                               ;CALL:
                                               ;        JSR     PC,OPRDAT
                                               ;        RETURN
                                               ;
                                               ;NOTE: THIS ROUTINE IS ENTERED ONLY AT INITIAL START

   076160  104401  076302                      OPRDAT: TYPE     ,ENTDAT          ;'ENTER DATE'
   076164  104411                                      RDLIN                     ;READ THE ENTRY
   076166  012605                                      MOV      (SP)+,R5         ;PUT THE ENTRY ADDRESS INTO R5
   076170  112537  001322                              MOVB     (R5)+,DATE       ;STORE THE DATE
   076174  112537  001323                              MOVB     (R5)+,DATE+1     ;STORE THE DATE
   076200  112537  001324                              MOVB     (R5)+,DATE+2     ;STORE THE DATE
   076204  112537  001325                              MOVB     (R5)+,DATE+3     ;STORE THE DATE
   076210  112537  001326                              MOVB     (R5)+,DATE+4     ;STORE THE DATE
   076214  112537  001327                              MOVB     (R5)+,DATE+5     ;STORE THE DATE
   076220  112537  001330                              MOVB     (R5)+,DATE+6     ;STORE THE DATE
   076224  112537  001331                              MOVB     (R5)+,DATE+7     ;STORE THE DATE
   076230  005037  001332                              CLR      DATE+8.          ;SET TERMINATOR
   076234  104401  076320                              TYPE     ,ENTID           ;'ENTER OPERATOR I.D.'
   076240  104411                                      RDLIN                     ;READ THE ENTRY
   076242  012605                                      MOV      (SP)+,R5         ;ENTRY ADDRESS
   076244  112537  001334                              MOVB     (R5)+,OPERID     ;STORE THE I.D.
   076250  112537  001335                              MOVB     (R5)+,OPERID+1   ;STORE THE I.D.
   076254  112537  001336                              MOVB     (R5)+,OPERID+2   ;STORE THE I.D.
   076260  112537  001337                              MOVB     (R5)+,OPERID+3   ;STORE THE I.D.
   076264  112537  001340                              MOVB     (R5)+,OPERID+4   ;STORE THE I.D.
   076270  112537  001341                              MOVB     (R5)+,OPERID+5   ;STORE THE I.D.
   076274  005037  001342                              CLR      OPERID+6         ;SET TERMINTOR
   076300  000207                                      RTS      PC               ;RETURN

   076302     200     105     116              ENTDAT: .ASCIZ   <CRLF>/ENTER DATE: /
   076320     105     116     124              ENTID:  .ASCIZ   /ENTER OPERATOR I.D.: /
                                               .EVEN

                                               .SBTTL  ROUTINE TO SIZE MEMORY

                                               ;;*******************************************************************
                                               ;*CALL:
                                               ;*       JSR     PC,$SIZE
                                               ;*       RETURN
                                               ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION

   076346  010046                              $SIZE:  MOV      R0,-(SP)          ;;SAVE R0 ON THE STACK
   076350  010146                                      MOV      R1,-(SP)          ;;SAVE R1 ON THE STACK
   076352  013746  000114                              MOV      @#114,-(SP)       ;;SAVE MEMORY ERROR VECTOR PS & PC
   076356  013746  000116                              MOV      @#116,-(SP)
   076362  012737  000116  000114                      MOV      #116,@#114        ;;IGNORE PARITY ERRORS WHILE SIZING
   076370  012737  000002  000116                      MOV      #RTI,@#116
   076376  013746  000004                              MOV      @#ERRVEC,-(SP)    ;;SAVE PRESENT ERROR VECTOR PS & PC
   076402  013746  000006                              MOV      @#ERRVEC+2,-(SP)
   076406  010600                                      MOV      SP,R0             ;;SAVE THE STACK POINTER
                                               ;;SET THE ERRVEC PS TO THE PRESENT PS
   076410  104400                                      TRAP                       ;;PUSH OLD PSW AND PC ON STACK
   076412  012637  000006                              MOV      (SP)+,@#ERRVEC+2          ;;SAVE THE PSW IN @#ERRVEC+2
   076416  012737  076436  000004                      MOV      #2$,@#ERRVEC      ;;SET FOR TIMEOUT
   076424  012701  020000                              MOV      #20000,R1         ;;FIRST ADDRESS
   076430  005711                              1$:     TST      (R1)              ;;TEST THIS ADDRESS
```

```
       076432  005721                          TST     (R1)+                   ;;STEP TO NEXT ADDRESS
       076434  000775                          BR      1$                      ;;TRY ANOTHER
       076436  162701  000002          2$:     SUB     #2,R1                   ;;DROP BACK
       076442  010006                          MOV     R0,SP                   ;;RESTORE THE STACK
       076444  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
       076450  012637  000004                  MOV     (SP)+,@#ERRVEC
       076454  012637  000116                  MOV     (SP)+,@#116             ;;RESTORE MEMORY ERROR VECTOR
       076460  012637  000114                  MOV     (SP)+,@#114
       076464  010137  076476                  MOV     R1,$LSTAD               ;;LAST ADDRESS
       076470  012601                          MOV     (SP)+,R1                ;;RESTORE R1
       076472  012600                          MOV     (SP)+,R0                ;;RESTORE R0
       076474  000207                          RTS     PC
       076476  000000                  $LSTAD: .WORD   0                       ;;CONTAINS THE LAST ADDRESS
  86
  87                                   .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH/RM
  88                                   ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
  89                                   ;OF THE RH/RM IS SETUP FOR THE PROPER ADDRESS.
  90                                   ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
  91                                   ;REQUIRED.
  92                                   ;NOTE: THIS ROUTINE DESTROYS R0-R4
  93                                   ;CALL
  94                                   ;
  95                                   ;       JSR     PC,BUSADR
  96                                   ;       RETURN
  97
  98   076500  005737  001360          BUSADR: TST     CHGADR                  ;INPUT FROM TTY REQUESTED?
  99   076504  002036                          BGE     3$                      ;NO--BRANCH
 100   076506  005037  001360                  CLR     CHGADR                  ;YES--CLEAR THE REQUEST FLAG
 101   076512  104401  001203                  TYPE    ,$CRLF                  ;TYPE A CR-LF
 102   076516  012700  001272          1$:     MOV     #$RMADR,R0              ;FIRST ADDRESS
 103   076522  104401  076620                  TYPE    ,MRMCS1                 ;'RMCS1='
 104   076526  011046                          MOV     (R0),-(SP)              ;PRESENT RMCS1 ADDRESS
 105   076530  104402                          TYPOC                           ;TYPE IT
 106   076532  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
 107   076536  104411                          RDLIN                           ;GET THE ENTRY
 108   076540  012601                          MOV     (SP)+,R1                ;ADDRESS OF ASCII TEXT
 109   076542  004537  076636                  JSR     R5,CK.NUM               ;ENTER AND STORE THE NEW ADDRESS
 110   076546  000763                          BR      1$                      ;ERROR EXIT
 111   076550  012700  001274          2$:     MOV     #$RMVEC,R0              ;VECTOR ADDRESS
 112   076554  104401  076627                  TYPE    ,MRMVEC                 ;'RMVEC='
 113   076560  011046                          MOV     (R0),-(SP)              ;PRESENT RH/RM VECTOR ADDRESS ON THE STACK
 114   076562  104402                          TYPOC                           ;TYPE IT
 115   076564  104401  073270                  TYPE    ,BLNKS2                 ;TYPE 2 BLANKS
 116   076570  104411                          RDLIN                           ;READ THE ENTRY
 117   076572  012601                          MOV     (SP)+,R1                ;ASCII TEXT ADDRESS
 118   076574  004537  076636                  JSR     R5,CK.NUM               ;ENTER AND STORE NEW ADDRESS
 119   076600  000763                          BR      2$                      ;ERROR EXIT
 120   076602  012700  001272          3$:     MOV     #$RMADR,R0              ;FIRST ADDRESS OF NEW PARAMETERS
 121   076606  012701  035762                  MOV     #RMADR,R1               ;FIRST ADDRESS OF WHERE TO PUT THEM
 122   076612  012021                          MOV     (R0)+,(R1)+             ;BUS ADDRESS
 123   076614  012021                          MOV     (R0)+,(R1)+             ;VECTOR ADDRESS
 124   076616  000207                          RTS     PC                      ;RETURN
 125
 126   076620     122     115     103  MRMCS1: .ASCIZ  @RMCS1=@
 127   076627     122     115     126  MRMVEC: .ASCIZ  @RMVEC=@
 128
 129                                   .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
```

```
                          ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
                          ;AND FORMS AN OCTAL NUMBER IN R2
                          ;CALL:
                          ;        MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                          ;        JSR     R5,CK.NUM       ;R5 CHANGED
                          ;        RET                     ;ERROR EXIT
                          ;        RET                     ;NORMAL EXIT

076636  010246            CK.NUM:  MOV     R2,-(SP)        ;SAVE R2
076640  010346                     MOV     R3,-(SP)        ;SAVE R3
076642  010446                     MOV     R4,-(SP)        ;SAVE R4
076644  012703  000006             MOV     #6,R3           ;MAX OCTAL DIGITS IN THE NUMBER
076650  005002                     CLR     R2              ;FINAL OCTAL VALUE
076652  112104            1$:      MOVB    (R1)+,R4        ;GET CURRENT POINTED BYTE
076654  001424                     BEQ     3$              ;BRANCH,IF TERMINATOR DETECTED
076656  120427  000060             CMPB    R4,#'0          ;SMALLER THAN ASCII-0 ?
076662  103425                     BLO     5$              ;YES,ERROR EXIT
076664  120427  000067             CMPB    R4,#'7          ;LARGER THAN ASCII-7 ?
076670  101022                     BHI     5$              ;YES,ERROR EXIT
076672  006302                     ASL     R2              ;SHIFT LEFT
076674  103420                     BCS     5$              ;
076676  006302                     ASL     R2              ;ONE
076700  103416                     BCS     5$              ;
076702  006302                     ASL     R2              ;OCTAL DIGIT
076704  103414                     BCS     5$              ;ERROR IF CARRY BIT SET
076706  042704  177770             BIC     #177770,R4      ;CHOP OFF HIGHER BITS
076712  060402                     ADD     R4,R2           ;APPENDING CURRENT DIGIT TO NUMBER
076714  005303                     DEC     R3              ;DECREMENT BYTE COUNT
076716  001401                     BEQ     2$              ;BRANCH,IF LAST BYTE
076720  000754                     BR      1$              ;LOOPING BACK
076722  112104            2$:      MOVB    (R1)+,R4        ;CHECK TERMINATOR
076724  001004                     BNE     5$              ;ERROR EXIT
076726  005702            3$:      TST     R2              ;FINAL VALUE = 0
076730  001401                     BEQ     4$              ;YES,THEN NOT REPLACE THE ORIGINAL VALUE
076732  010210                     MOV     R2,(R0)         ;REPLACE THE ORIGINAL VALUE
076734  005725            4$:      TST     (R5)+           ;ADJUST FOR NORMAL RETURN
076736  012604            5$:      MOV     (SP)+,R4        ;RESTORE R4
076740  012603                     MOV     (SP)+,R3        ;RESTORE R3
076742  012602                     MOV     (SP)+,R2        ;RESTORE R2
076744  000205                     RTS     R5              ;EXIT

076746                    CYLNDR:  .BLKW   258.            ;ONE SECTOR WORD CTR MAX SIZE

        077752            ENDPGM=.

        000200            .END     200
```

```
ABASE = 176700    ASGN7   025532    BIT8  = 000400    CR    = 000015    DROPNG  074011
ABNRML  030152    ASKPAR  075222    BIT9  = 001000    CRLF  = 000200    DRQ   = 004000
ACDW1 = 000000    ASNERR  030026    BLKADR  002106    CYLIMT  001444    DRVACT  035624
ACDW2 = 000000    ASNLST  001550    BLNKS1  073271    CYLNDR  076746    DRVCLR= 000111
ACK   = 000123    ASNMSG  030052    BLNKS2  073270    DATAPK  026132    DRVER   011322
ACPUOP= 000000    ASSIGN  025166    BLNKS3  073267    DATA0   002546    DRVINT  036212
ACTDRV  035700    ASWREG= 000000    BLNKS4  073266    DATA1   002606    DRVPRM  026374
ACTSTR  035701    ATA   = 100000    BPTVEC= 000014    DATA10  003246    DRVQUE  044044
ADDW0 = 000000    ATABIT  035750    BUFTBL  001704    DATA11  003306    DRVSTA  035634
ADDW1 = 000000    ATESTN= 000000    BUSADR  076500    DATA12  003346    DRVTYP  035644
ADDW10= 000000    ATTN    001344    BUSY    074567    DATA13  003406    DRY   = 000200
ADDW11= 000000    AT0   = 000001    CFLAG   001362    DATA14  003446    DSWR  = 177570
ADDW12= 000000    AT1   = 000002    CHGADR  001360    DATA15  003506    DTE   = 010000
ADDW13= 000000    AT2   = 000004    CHKADR  017656    DATA2   002646    DTEER   012160
ADDW14= 000000    AT3   = 000010    CI1     037300    DATA3   002706    DTUW    035746
ADDW15= 000000    AT4   = 000020    CI3     037406    DATA4   002746    DT00  = 000001
ADDW2 = 000000    AT5   = 000040    CI4     037506    DATA5   003006    DT01  = 000002
ADDW3 = 000000    AT6   = 000100    CI5     040064    DATA6   003046    DT02  = 000004
ADDW4 = 000000    AT7   = 000200    CI6     040106    DATA7   003106    DT03  = 000010
ADDW5 = 000000    AUNIT = 000000    CI7     040122    DATA8   003146    DT04  = 000020
ADDW6 = 000000    AUSWR = 000000    CI7B    040144    DATA9   003206    DT05  = 000040
ADDW7 = 000000    AUTOCK  001506    CI8     040222    DATE    001322    DT06  = 000100
ADDW8 = 000000    AVAIL   001616    CKBUS   013400    DATEIS  074466    DT07  = 000200
ADDW9 = 000000    AVECT1= 000254    CKCLK   023326    DCK   = 100000    DT08  = 000400
ADEVCT= 000000    AVECT2= 000000    CKCLK1  023422    DCKER   010154    DT1     071244
ADEVM = 000000    A16   = 000400    CKCLK2  023470    DCKER1  010274    DT14    071314
AENV  = 000000    A17   = 001000    CKCLK3  023520    DDISP = 177570    DT15    071336
AENVM = 000000    BADENT  074546    CKERR   013306    DEASGN  025604    DT16    071360
AFATAL= 000000    BADSEC  001364    CKFMT   011354    DEASSG  074114    DT2     071250
AMADR1= 000000    BAI   = 000010    CKHCE   011550    DH1     070613    DT3     071266
AMADR2= 000000    BEGCOD  001522    CKSWR = 104407    DH14    070770    DT4     071276
AMADR3= 000000    BEGPAT  001520    CK.CHR  031314    DH15    071073    DT6     071310
AMADR4= 000000    BEGSIZ  001524    CK.DEC  031266    DH16    071172    DULP    036214
AMAMS1= 000000    BIT0  = 000001    CK.DIG  031366    DH2     070620    DUNIT   001552
AMAMS2= 000000    BIT00 = 000001    CK.NUM  076636    DH3     070674    DVA   = 004000
AMAMS3= 000000    BIT01 = 000002    CK.OCT  031240    DH4     070722    DVC   = 000200
AMAMS4= 000000    BIT02 = 000004    CLKFLG  001312    DH6     070761    ECBAD0  001434
AMSGAD= 000000    BIT03 = 000010    CLOCK   024462    DISPLA  001156    ECBAD1  001442
AMSGLG= 000000    BIT04 = 000020    CLR   = 000040    DISPLY= 104414    ECBIT   001420
AMSGTY= 000000    BIT05 = 000040    CLRDPB  026174    DISPRE  000174    ECC     014710
AMTYP1= 000000    BIT06 = 000100    CLRQUE  043746    DLT   = 100000    ECCX    015442
AMTYP2= 000000    BIT07 = 000200    CMCNT   001412    DONE    007646    ECC1    015306
AMTYP3= 000000    BIT08 = 000400    CMCYL   001414    DPE   = 020000    ECC2    015436
AMTYP4= 000000    BIT09 = 001000    CMDAT   013736    DPINT   035654    ECGD    001432
AOE   = 001000    BIT1  = 000002    CMHED   013664    DPR   = 000400    ECGD1   001440
APASS = 000000    BIT10 = 002000    CMPAR   013464    DPRQS   035664    ECH   = 000100
APRIOR= 000000    BIT11 = 004000    CMPARD  013502    DRIVE = 001220    ECI   = 004000
APTCSU= 000040    BIT12 = 010000    CMPLMT  001476    DRIVE0  044210    ECMSK0  001424
APTENV= 000001    BIT13 = 020000    CMPRES  020356    DRIVE1  046376    ECMSK1  001426
APTSIZ= 000200    BIT14 = 040000    CMPRT   014310    DRIVE2  050564    ECSEC   001422
APTSPO= 000100    BIT15 = 100000    CMPRX   014302    DRIVE3  052752    ECWRD   001430
ASGND   074162    BIT2  = 000004    CMSEC   001416    DRIVE4  055140    ECWRD1  001436
ASGN1   025174    BIT3  = 000010    CMSTR   013602    DRIVE5  057326    EMPTYQ  044024
ASGN2   025260    BIT4  = 000020    CMSTR2  013724    DRIVE6  061514    EMTVEC= 000030
ASGN3   025356    BIT5  = 000040    CMTRK   001417    DRIVE7  063702    EM1     066160
ASGN4   025516    BIT6  = 000100    COLON   074464    DRNUM   074136    EM10    066464
ASGN6   025520    BIT7  = 000200    COMTBL  002126    DROP    030056    EM11    066527
```

```
EM12    066562    ERRVEC= 000004    INTDON  074610    LINP5   072057    MDPE  = 000400
EM13    066613    FACTOR  016310    INTRVL  001472    LINR6   072254    MERR1   074716
EM14    066665    FAIRNS  001354    INVLD   074307    LINSS3  071723    MERR2   074774
EM15    066710    FALPAR  007432    IOTVEC= 000020    LINST3  071706    MINCYL= 000110
EM2     066230    FALPR1  007446    IR    = 000100    LINS3   071560    MINSEC= 000120
EM20    066744    FER   = 000020    ISR     040634    LINS4   071757    MINTRK= 000114
EM21    066765    FILBUF  016676    IVC   = 010000    LINS5   072075    MINUTE  001370
EM22    067016    FISK    030552    KSR     024624    LINT3   071632    MNDLTA  035774
EM23    067071    FMTER   012354    KSR1    024636    LINU06  072265    MNTBL   002156
EM24    067150    FMT16 = 010000    LA      040460    LINW3   071674    MOH   = 020000
EM25    067216    FORMAT  001500    LACNT   035712    LINX4   071771    MOL   = 010000
EM26    067277    FOWT1   005644    LBC   = 002000    LINX5   073273    MONTR   005614
EM27    067324    FOWT2   005720    LBT   = 002000    LIN10A  072704    MRMCS1  076620
EM3     066266    FOWT3   006070    LF    = 000012    LIN10B  072740    MRMVEC  076627
EM30    067361    FRSTER  001400    LIMIT   001410    LIN10C  073001    MSFULL  075040
EM31    067413    F0    = 000002    LINA3   071612    LIN10H  073106    MSGCYL  075072
EM32    067456    F1    = 000004    LINB3   071664    LIN11H  073173    MSGFOR  074107
EM33    067511    F2    = 000010    LINB5   072041    LIN2C   071374    MSGON   074064
EM34    067566    F3    = 000020    LINB6   072154    LIN2P   071415    MSGSEC  075110
EM35    067630    F4    = 000040    LINCA3  071643    LIN2S   071441    MSGTRK  075101
EM36    067666    GENDPB  066070    LINC6   072207    LIN3.1  022040    MSWRO   074252
EM37    067717    GENREG  066110    LINDA3  071654    LIN3.3  022146    MXDLTA  035772
EM4     066324    GETADR  026700    LINDEC  023156    LIN3.4  022200    MXF   = 001000
EM40    070001    GETBUF  016312    LIND5   072024    LIN6.1  022606    MXLACT  035770
EM41    070044    GETID   026604    LINEN3  071575    LIN6.2  022630    MXWNDW  035776
EM42    070110    GETLMT  026546    LINEO5  072142    LIN7M   072312    M.DPID  030656
EM43    070171    GETPAR  020064    LINEP5  072131    LIN7O   072340    M.DP40  030714
EM44    070236    GETPAT  020022    LINE1   021112    LIN7P   072350    M.DP41  030750
EM45    070327    GETREG= 000141    LINE2   021166    LIN7R   072440    M.DP42  030760
EM46    070415    GETREM  030562    LINE2A  021336    LIN7S   072370    M.DP44  031012
EM5     066361    GETREQ  044120    LINE2B  021354    LIN7T   072412    M.DP50  031024
EM50    070467    GO    = 000001    LINE3   021574    LIN7X   072424    N       074244
EM51    070525    GODRIV  017024    LINE3A  021602    LIN8M   072455    NED   = 010000
EM6     066415    GTSWR = 104406    LINE3B  021610    LIN9B   072500    NEDCLK  074173
EM60    070570    HCE   = 000200    LINE3C  021622    LIN9E   072625    NEM   = 004000
ENDCMP  014564    HCEER   012432    LINE3D  021632    LIN9G   072655    NEWASN  025574
ENDCON  001454    HCI   = 002000    LINE3E  021700    LIN9H   072527    NEWUNT  001574
ENDET   001512    HCRC  = 000400    LINE3F  021766    LIN9I   072605    NOMTCH  013034
ENDPAS  074045    HCRCER  011156    LINE4   022244    LKPAR   005232    NONE    074540
ENDPGM= 077752    HEADER  001516    LINE5   022334    LODEV   073515    NOTAVL  073466
ENDSEK  001460    HEDLIN  074517    LINE5A  022434    LSC   = 004000    NOTPRS  073451
ENDTST  074071    HOUR    001366    LINE5B  022502    LSTAD   001356    NOTPRT  001510
ENTADR  074427    HT    = 000011    LINE6   022544    MAIN    006100    NOTRM   073430
ENTCOM  074332    HZ      001314    LINE6A  022556    MAINDA  006126    NOTSAF  073505
ENTDAT  076302    IAE   = 002000    LINE6B  022564    MAIN1   006260    NSA   = 100000
ENTDRV  074353    IAEER   012274    LINE6C  022572    MAIN2   006400    OFFCOD  002470
ENTID   076320    IDIS    074476    LINE6D  022600    MAIN3   006500    OFFDIR= 000001
ENTLMT  074377    IDLE    006676    LINE7   022654    MANTER  027226    OFFON = 000001
ENTPR   005322    IDLE1   006774    LINE7A  022776    MASK    001350    OFFSET= 000115
EOPX    030476    IE    = 000100    LINE8   023112    MATCH   014632    OFFST   015710
EOP1    030232    ILF   = 000001    LING6   072236    MAXCYL= 000106    OFLIN   007544
EOP2    030254    ILR   = 000002    LINKDV  030610    MAXDL   001466    OFMSG0  002366
ERCTR   001406    INCHRD  024244    LINM3   071475    MAXER   001470    OFMSG1  002421
ERPRC1  007266    INCMIS  024314    LINM4   071740    MAXSEC= 000116    OFMSG2  002444
ERPROC  007252    INCSKI  024270    LINN3   071513    MAXTRK= 000112    OFMTBL  002474
ERR   = 040000    INCSOF  024220    LINOCT  023124    MCPE  = 020000    ONES    003450
ERROR = 104000    INCTOT  024340    LINP3   071537    MCPEMX  035760    OPE   = 000010
```

```
OPERID  001334     PSW   = 177776     RMCS1 = 000000     SEEK  = 000105     SW2   = 000004
OPI   = 020000     PUNSAF  007350     RMCS2 = 000010     SEEKFG  035724     SW3   = 000010
OPIER   012050     PWRVEC= 000024     RMDA  = 000006     SELDRV= 000145     SW4   = 000020
OPIER1  012114     QCNT    043454     RMDB  = 000022     SELPAR  017102     SW5   = 000040
OPRDAT  076160     QDRV0   043546     RMDC  = 000034     SETFMT= 000143     SW6   = 000100
OPT     037042     QDRV1   043566     RMDS  = 000012     SETVEC  005350     SW7   = 000200
OPTBL   002134     QDRV2   043606     RMDT  = 000026     SET.IE  043402     SW8   = 000400
OR    = 000200     QDRV3   043626     RMEC1 = 000044     SHDTYP  023644     SW9   = 001000
ORDERQ  001526     QDRV4   043646     RMEC2 = 000046     SIXTEE  001374     SYSTAT  073535
PACK    001320     QDRV5   043666     RMERRS  035614     SIZMEM  005054     T       071510
PAR   = 000010     QDRV6   043706     RMER1 = 000014     SKI   = 040000     TABLE   075460
PARENT  027702     QDRV7   043726     RMER2 = 000042     SKIER   012614     TABLE0  075500
PARER   012202     QINPT   043464     RMHR  = 000036     SLASH   075216     TABLE1  075546
PARLST  075120     QOUTPT  043504     RMINIT  036000     SPOTCK  020760     TABLE2  075614
PARQ    001662     QSTART  043524     RMLA  = 000020     SRCHWT  035676     TABLE3  075662
PAR1    075250     QSTOP   043526     RMMR1 = 000024     STACK = 001100     TABLE4  075730
PAR10   075360     QTERM = 043746     RMMR2 = 000040     START   003626     TABLE5  075776
PAR11   075370     QUES    073327     RMOF  = 000032     START1  003636     TABLE6  076044
PAR14   075400     RANCYL  017402     RMR   = 000004     START2  003654     TABLE7  076112
PAR15   075410     RANPAT  017636     RMSN  = 000030     STATHD  073644     TAB.XY= 001114
PAR16   075420     RANSEC  017242     RMTMR   042310     STATIN  001316     TAP   = 040000
PAR19   075430     RANSIZ  017516     RMVEC   035764     STATIS  016154     TBITVE= 000014
PAR2    075260     RANTRK  017316     RMWC  = 000002     STATPR  023530     TD      040700
PAR20   075440     RANWRT  017766     RM05    036550     STKLMT= 177774     THEAD   017162
PAR21   075450     RANXIT  017646     RNOP  = 000101     STNDAT  002502     TIMER   035726
PAR3    075270     RATIO   001504     RTC   = 000117     ST0     042400     TKVEC = 000060
PAR4    075300     RDCHR = 104410     RTNCTR  015622     ST01    042434     TPVEC = 000064
PAR5    075310     RDDAT = 000171     R6    =%000006     ST02    042530     TRAPVE= 000034
PAR6    075320     RDHD  = 000173     R7    =%000007     ST03    042560     TRE   = 040000
PAR7    075330     RDLIN = 104411     S       071534     ST05    042604     TRFER   012514
PAR8    075340     RDY   = 000200     SATPOW  033572     ST06    042612     TRKLMT  001450
PAR9    075350     RD.ADR  042746     SAVEFG  035722     ST07    042650     TRNSWT  035674
PASCNT  001464     RD.RM   042722     SAVER1  001402     ST08    042700     TRTVEC= 000014
PAT   = 000020     RD.RM1  042744     SAVER5  001404     ST09    042710     TST1    003644
PATTEN  001514     RD.RM2  043064     SAVREG= 104412     SUMARY  023616     TYPDS = 104405
PCLOCK  001310     RD.RM3  043070     SC      041154     SVRH70  043264     TYPE  = 104401
PERIOD  074250     RD.RM4  043074     SCMND   025704     SWR     001154     TYPOC = 104402
PGE   = 002000     RD.WRD  042750     SCOPE = 000004     SWREG   000176     TYPON = 104404
PGM   = 001000     READDR  023176     SC04  = 000400     SWTIM   007506     TYPOS = 104403
PIP   = 020000     READHD  015734     SC1   = 000100     SW0   = 000001     TYPRI4  031164
PIRQ  = 177772     READIN= 000121     SC10  = 001000     SW00  = 000001     UCPAR   007414
PIRQVE= 000240     RECAL = 000107     SC11    041772     SW01  = 000002     ULDFLG  035702
POPQUE  044142     RECALT  015646     SC12    042102     SW02  = 000004     UNIT    001346
POSER   011774     RECAL0  015652     SC13    042152     SW03  = 000010     UNS   = 040000
PROCES  007162     REDAPK  026162     SC2   = 000200     SW04  = 000020     UNSAF   012754
PRTBAD  015450     REFMT   007040     SC20  = 002000     SW05  = 000040     UNTASN  073402
PRTIM   007610     REFMTX  007160     SC3     041220     SW06  = 000100     UNTMSG  073331
PR0   = 000000     RELBUF  016446     SC4     041224     SW07  = 000200     UNTNOT  073360
PR1   = 000040     RELSE = 000113     SC5     041236     SW08  = 000400     UNTOFF  073337
PR2   = 000100     REPHD   073572     SC6     041422     SW09  = 001000     UNTON   073350
PR3   = 000140     REPLZ   031034     SC6A    041536     SW1   = 000002     UPE   = 020000
PR4   = 000200     RESREG= 104413     SC7     041664     SW10  = 002000     US1   = 000001
PR5   = 000240     RESVEC= 000010     SC8     041742     SW11  = 004000     US2   = 000002
PR6   = 000300     RETRY   001352     SDETAL  023666     SW12  = 010000     US4   = 000004
PR7   = 000340     RMADR   035762     SEARCH= 000131     SW13  = 020000     VV    = 000100
PS    = 177776     RMAS  = 000016     SECLMT  001446     SW14  = 040000     WAIT    001640
PSEL  = 002000     RMBA  = 000004     SECOND  001372     SW15  = 100000     WATPAK  026144
```

```
WCE    = 040000      $CNTLC   032256      $ITEMB   001130      $POSIT= 000042      $SKI   = 000064
WCF    = 000040      $CNTLG   034765      $LF      001204      $POWER  033562      $SOFT  = 000060
WCFER    012656      $CNTLU   034760      $LFLG    033405      $PREVA= 000032      $SSEC  = 000022
WCHKX  = 000001      $CODE  = 000024      $LKCSB   001300      $PREVO= 000027      $STUP  = 177777
WCKD   = 000151      $COMND= 000002       $LKCSR   001276      $PSEL  = 000003     $SUPRS   031124
WCKER    010630      $CPUOP   001234      $LKS     001304      $PWRAD  033550      $SVPC  = 000210
WCKHD  = 000153      $CRLF    001203      $LLVEC   001306      $PWRDN  033410      $SWR   = 122000
WCSEL    001502      $CYL   = 000012      $LONUM   035114      $PWRMG  033544      $SWREG   001230
WC.HK    041054      $DBLK    034346      $LPADR   001122      $PWRUP  033462      $TATUS=  000016
WLE    = 004000      $DB2D    035212      $LPERR   001124      $QUES   001202      $TERM  = 000032
WLEER    012326      $DB2O    035406      $LPVEC   001302      $RAND   035014      $TESTN   001212
WRL    = 004000      $DECVL   035372      $LSTAD   076476      $RDCHR  034640      $TIME    024364
WRTDAT=  000161      $DEVCT   001216      $MADR1   001240      $RDLIN  031766      $TKB     001162
WRTHD  = 000163      $DEVM    001264      $MADR2   001244      $RDSZ  = 000001     $TKINT   031604
WRTPK    020374      $DOAGN   030546      $MADR3   001250      $READ  = 000052     $TKS     001160
WRTPK1   020424      $DRVID=  002106      $MADR4   001254      $REG   = 000014     $TKCRV   031634
WRTPK2   020656      $DSPLY   031212      $MAIL    001206      $RESRE  035154      $TMPO    001174
WRTPK3   020702      $DTBL    034336      $MAMS1   001236      $RETRY  016026      $TN    = 000002
WRTPK4   020734      $ENDAD   030536      $MAMS2   001242      $RMADR  001272      $TNPWR   035322
WRTPK5   020742      $ENDCT   030522      $MAMS3   001246      $RMAS  = 002134     $TOTAL=  000056
WRT.AD   043166      $ENV     001226      $MAMS4   001252      $RMBA  = 002122     $TPB     001166
WRT.RM   043076      $ENVM    001227      $MBADR   001102      $RMCS1= 002116      $TPFLG   001173
WRT.R1   043162      $EOP     030200      $MFLG    033404      $RMCS2= 002126      $TPS     001164
WRT.R2   043250      $EOPCT   030514      $MISPO=  000066      $RMDA  = 002124     $TRANS=  000046
WRT.R3   043254      $ERFLG   001117      $MNEW    035003      $RMDB  = 002140     $TRAP    035526
WRT.R4   043260      $ERMAX   001131      $MSGAD   001222      $RMDC  = 002152     $TRAP2   035550
WRT.R5   043262      $ERROR   032262      $MSGLG   001224      $RMDS  = 002130     $TRK   = 000011
WRT.WD   043164      $ERRPC   001132      $MSGTY   001206      $RMDT  = 002144     $TRP   = 000015
XWCNT    017456      $ERRTB   003546      $MSWR    034772      $RMEC1= 002162      $TRPAD   035562
XXDP     001452      $ERRTY   032452      $MTYP1   001237      $RMEC2= 002164      $TSTM    001104
Y        074246      $ERTTL   001126      $MTYP2   001243      $RMER1= 002132      $TSTNM   001116
ZEROS    002546      $ETABL   001226      $MTYP3   001247      $RMER2= 002160      $TTYIN   032244
ZROIND   001376      $ETEND   001272      $MTYP4   001253      $RMHR  = 002154     $TYPDS   034132
$APTHD   001100      $FAIR  = 000072      $NCODE=  000074      $RMLA  = 002136     $TYPE    032606
$ATYC    033166      $FATAL   001210      $NCYL  = 000100      $RMMR1= 002142      $TYPEC   033020
$ATY1    033142      $FFLG    033406      $NEXT  = 000104      $RMMR2= 002156      $TYPEX   033140
$ATY3    033150      $FILLC   001172      $NPATC=  000075      $RMOF  = 002150     $TYPOC   033730
$ATY4    033160      $FILLS   001171      $NSEC  = 000076      $RMSN  = 002146     $TYPON   033744
$AUTOB   001150      $FIRST=  000122      $NTRK  = 000077      $RMVEC  001274      $TYPOS   033704
$BASE    001262      $FMT   = 000001      $NULL    001170      $RMWC  = 002120     $UNIT    001220
$BDADR   001136      $GDADR   001134      $NWRDL=  000102      $RM02   073553      $UNITM   001110
$BDDAT   001142      $GDDAT   001140      $NWTST=  000000      $RM03   073560      $USWR    001232
$BDSEC=  000124      $GET42   030526      $OCNT    034126      $RM05   073565      $VECT1   001256
$BELL    001176      $GTSWR   034426      $OCTVL   035510      $RTNAD  030550      $VECT2   001260
$BUF   = 000006      $HARD  = 000062      $OMODE   034130      $SAVRE  035116      $WRDL  = 000020
$CDW1    001266      $HD    = 000000      $OPERC=  000036      $SAVR6  033560      $WRDM  = 000004
$CDW2    001270      $HIBTS   001100      $PACK  = 000026      $SB2D   031524      $XOFF  = 000023
$CHARC   033136      $HINUM   035112      $PASS    001214      $SB2O   031554      $XON   = 000021
$CKSWR   034356      $ICNT    001120      $PASSC=  000070      $SEC   = 000010     $$GET4=  000000
$CMTAG   001114      $ILLUP   033554      $PASTM   001106      $SETUP= 000156      $OFILL   034127
$CM3   = 000000      $INTAG   001151      $PATTC=  000030      $SIZE   076346      .$X    = 001100
$CM4   = 000001

. ABS.   077752     000
         000000     001
ERRORS DETECTED:  0
```

```
VIRTUAL MEMORY USED:  53264 WORDS  ( 209 PAGES)
```

```
DYNAMIC MEMORY AVAILABLE FOR  69 PAGES
CZRMUA.BIC,CZRMUA/C=CZRMUA.DOC,CZRMUA,SYSMAC/M
```

```
$$GET4    9-P41     9-P41#
$OFILL    9-U01     9-U01#    9-U01*    9-U01*
$40CAT    9-T68
$APTHD    4-275     4-275#
$ASTAT    9-T74     9-T74
$ATY1     9-T74#
$ATY3     9-T72     9-T74#
$ATY4     9-T68     9-T74#
$ATYC     9-T74     9-T74#
$AUTOB    5-0#      8-14*     8-51*     8-64      8-117     8-201     9-N06     9-U05     9-U05     9-U05
$BASE     5-0#      8-57      8-59
$BDADR    5-0#
$BDDAT    5-0#
$BDSEC    9-@42     9-B31     9-M38     9-N12     10->75#   10->79
$BELL     5-0#      9-B73     9-T68     9-T68     9-T68
$BUF      9-85*     9-118*    9-869     9-891     9-=25     9->04     9->10     9->15     9->19     9->61     9-C51     9-E01     9-E08     10->25#
$CDW1     5-0#      8-119     9-T98*
$CDW2     5-0#      9-M51*    9-M60     9-M80*
$CHARC    9-T72     9-T72#    9-T72*    9-T72*    9-T72*
$CKSWR    9-U05#    9-U15     9-U15
$CM3      5-0       5-0#
$CM4      5-0       5-0       5-0#      5-0#
$CMTAG    5-0#      8-13      8-13      8-13
$CNTLC    9-S70     9-T29     9-T54#
$CNTLG    9-S79     9-U05     9-U05#
$CNTLU    9-T24     9-U05     9-U05#
$CODE     9-834     9-883     9-910     9-962     9-;01     9-;87     9-=30     9-=32     9->59     9->63     9-@67     9-A12*    9-A20*    9-A35
          9-A86*    9-B07*    9-B11*    9-L36*    9-L45     10-791    10-793    10-801*   10->36#
$COMND    9-<14*    9-<32*    9-<50*    9-<69*    9-A14*    9-A15     9-A19*    9-A22     9-B08*    9-B12*    9-L38*    9-M50*    10-802*   10-803*
         10->22#
$CPUOP    5-0#
$CRLF     5-0#      8-34      8-129     8-132     8-152     8-196     8-206     9-37      9-58      9-248     9-260     9-261     9-267     9-268
          9-293     9-822     9-831     9-831     9-831     9-831     9-:27     9-:47     9-:59     9-;60     9-;68     9-<02     9-<04     9-B76
          9-B77     9-B79     9-B95     9-C28     9-C30     9-C55     9-C68     9-C87     9-D07     9-D26     9-D41     9-D69     9-D92     9-E11
          9-E32     9-E46     9-E59     9-E66     9-E99     9-F08     9-F32     9-F57     9-G89     9-H21     9-I09     9-J87     9-K87     9-M89
          9-027     9-045     9-P41     9-P41     9-S69     9-S78     9-T34     9-T68     9-T68     9-T68     9-T68     9-T70     9-T70     9-T70
          9-T70     9-T72     9-T72     9-T72     9-U05     16-101
$CYL      9-894     9-<67*    9-=38     9->65     9-?00     9-A00     9-A30*    9-A77*    9-A78     9-A85*    9-D14     9-L81*    9-M46*    10-799
         10->28#
$DB2D     9-F17     9-F25     9-F30     9-F44     9-F49     9-H49     9-H49     9-H57     9-H65     9-S28     9-U11#
$DB2O     9-S45     9-U13#
$DBLK     9-U03     9-U03     9-U03#
$DECVL    9-U11     9-U11#
$DEVCT    5-0#      9-P41*
$DEVM     5-0#
$DOAGN    9-P41     9-P41     9-P41#
$DRVID    9-K76     9-M17*    9-M20*    9-M20*    9-M20*    9-M20*    9-M20*    10->79#   10->83
$DSPLY    9-Q96#    9-U16
$DTBL     9-U03     9-U03#
$ENDAD    4-272     9-P41#
$ENDCT    9-P41#
$ENV      5-0#      8-14      8-51      8-52      8-121     8-212     9-T68     9-T72     9-T74     9-T74
$ENVM     5-0#      8-13      9-T72     9-T72     9-T74
$EOP      9-162     9-P41#
$EOPCT    9-P41     9-P41#
$ERFLG    5-0#      9-T68     9-T68     9-T68*
```

```
$ERMAX   5-0#
$ERROR   8-13      9-T68#
$ERRPC   5-0#      9-T68     9-T68     9-T68     9-T68*    9-T68*    9-T70
$ERRTB   7-0#      9-T70
$ERRTY   9-T68     9-T70#
$ERTTL   5-0#      9-T68     9-T68     9-T68*
$ESCAP  10-817    10-868    10-870    10-;32    10-;91    10->00
$ETABL   5-0#
$ETEND   4-275     5-0#
$FAIR    9-87*     9-121*    9-129*    9-130    10->51#
$FATAL   5-0#      9-T74*
$FFLG    9-T74     9-T74#    9-T74*    9-T74*    9-T74*
$FILLC   5-0#      9-T72     9-T72     9-T72
$FILLS   5-0#      9-T72     9-T72
$FIRST   9-L64*   10->71#
$FMT     9-423*   10->21#   10->22    10->23    10->24    10->25    10->26    10->27    10->28    10->29    10->30    10->34
$GDADR   5-0#
$GDDAT   5-0#
$GET42   9-20      9-G66     9-056     9-P41#
$GTSWR   9-S83     9-U05#    9-U15     9-U15
$HARD    9-H77     9-I00     9-I00*   10->47#
$HD      4-63      4-63      4-63
$HIBTS   4-275#
$HINUM   8-13*     9-P52     9-U07     9-U07     9-U07#    9-U07*
$ICNT    5-0#
$ILLUP   9-T76     9-T76     9-T76#
$INTAG   5-0#      9-U05     9-U05     9-U05
$ITEMB   5-0#      9-T68     9-T68     9-T68     9-T68*    9-T70
$LF      5-0#      9-T44     9-T68     9-T68     9-T72     9-T72
$LFLG    9-T74#    9-T74*
$LKCSB   6-0#      9-G50*
$LKCSR   6-0#      9-G44     9-G51*
$LKS     6-0#      9-G55     9-G60*
$LLVEC   6-0#      9-G57
$LONUM   8-13*     9-?21     9-P51     9-U07     9-U07     9-U07#    9-U07*
$LPADR   5-0#
$LPERR   5-0#
$LPVEC   6-0#      9-G47
$LSTAD   8-93     16-85#    16-85*
$MADR1   5-0#
$MADR2   5-0#
$MADR3   5-0#
$MADR4   5-0#
$MAIL    4-275     4-275     5-0#      8-7       8-13      8-14      8-51      9-T68     9-T72
$MAMS1   5-0#
$MAMS2   5-0#
$MAMS3   5-0#
$MAMS4   5-0#
$MBADR   4-275#
$MFLG    9-T74     9-T74#    9-T74*    9-T74*
$MISPO   9-F52     9-H77     9-I02     9-I02*   10->49#
$MNEW    9-U05     9-U05#
$MSGAD   5-0#      9-T74     9-T74*
$MSGLG   5-0#      9-T74*
$MSGTY   5-0#      9-T74     9-T74     9-T74*    9-T74*
$MSWR    9-U05     9-U05#
```

```
$MTYP2   5-0#
$MTYP3   5-0#
$MTYP4   5-0#
$NCODE   9-197*    9-?24*    9-?87     9-?90     9-@15     9-@22     9-@70*    9-@98     9-A12     9-A13     10->55#   10->56    10->57    10->58
         10->59    10->60    10->61    10->65
$NCYL    9-193*    9-?28*    9-?29     9-?31*    9-?85*    9-@45     9-A30     10->59#
$NEXT    9-108     9-203*    9-@28*    9-A38*    9-B15*    9-L27     10->61#
$NPATC   9-202*    9-@27*    9-A28     10->56#
$NSEC    9-195*    9-?27*    9-?32*    9-?50*    9-A29     10->57#
$NTRK    9-194*    9-?33*    9-?69*    10->58#
$NULL    5-0#      9-T72     9-T72     9-T72
$NWRDL   9-196*    9-?89*    9-@21*    9-A31     9-A32     10->60#
$NWTST   8-7#
$OCNT    9-U01#    9-U01*    9-U01*
$OCTVL   9-U13     9-U13#
$OMODE   9-U01     9-U01#    9-U01*    9-U01*    9-U01*    9-U01*
$OPERC   9->98*    9->99*    9-A64     9-A66     9-F15     9-F42     9-H34     9-P41     10->41#
$PACK    8-240*    9-110     9-A68     9-A87     9-B04     9-B18*    9-J98*    10->37#
$PASS    5-0#      8-13*     9-P41     9-P41     9-P41*    9-P41*    9-P41*    9-P41*
$PASSC   9-H37     9-J95*    9-P41     9-P41     9-P41*    10->50#
$PASTM   4-275#
$PATTC   9->72     9-A28*    9-B14*    9-L39*    9-L40*    10->39#
$POSIT   9-=40*    9-=41*    9-?02*    9-?03*    9-F47     9-P41     9-P41     10->42#
$POWER   9-T76     9-T76#
$PREVA   9-?00     9-A00*    9-A01*    9-A04*    9-A05*    9-A06*    9-A73*    9-A74*    9-A75*    9-D59     9-D63     9-D67     9-D75     10-799*
         10-800*   10->40#
$PREVO   9-@96*    9-@97*    9-A17     9-A24     9-A71*    9-C05     10-798*   10->38#
$PSEL    10->23#
$PWRAD   9-T76#
$PWRDN   8-13      9-T76     9-T76#
$PWRMG   9-T76#
$PWRUP   9-T76     9-T76#
$QUES    5-0#      9-T37     9-T68     9-T68     9-T72     9-T72     9-U05
$R2A     9-U15
$RAND    9-?12     9-@06     9-@83     9-A63     9-U07#
$RDCHR   9-U05#    9-U15     9-U15
$RDDEC   9-U15
$RDLIN   9-S97#    9-U15     9-U15
$RDOCT   9-U15
$RDSZ    9-U05     9-U05#
$READ    9-=36*    9-=37*    9-F28     9-P41     9-P41     10->44#
$REG     9-L27     10->29#   10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11
         10-?11    10-?11    10-?11    10-?11
$RESRE   9-U09#    9-U15
$RETRY   9-382     9-467     9-534     9-581     9-622     9-658     9-686     9-756     9-787     9-803     9-<89#    9-=07
$RM02    8-185     15-15#
$RM03    8-182     15-16#
$RM05    8-188     15-17#
$RMADR   6-0#      8-59*     8-60      8-67      13-5      16-102    16-120
$RMAS    10->90#   13-7
$RMBA    9-870     9-:91     9-;22     9-;43     9-;75     9-;95     9-=24     9-C52     9-D02     9-E07     9-E18*    9-E19     10->85#   13-7
$RMCS1   9-211     9-213     9-846     9-@96     9-A71     9-B99     10-798    10->83#   10->84    10->85    10->86    10->87    10->88    10->89
         10->90    10->91    10->92    10->93    10->94    10->95    10->96    10->97    10->98    10->99    10-?00    10-?01    10-?02    10-?11
         10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    13-6
$RMCS2   9-319     9-477     9-527     9-573     9-614     9-720     9-735     9-752     9-848     10->87#   13-6
$RMDA    9-<33*    9-?27     9-A76     9-D35     9-D90     9-G19     9-G20     10->86#   13-7
```

```
$RMDC     9-190     9-601     9-<34*    9-=38     9-?28     9-A77     9-D39     9-D78     9-G21    10->97#   13-8
$RMDS     9-215     9-322     9-351    10->88#   13-6
$RMDT    10->94#   13-7
$RMEC1    9-370     9-372     9-;07     9-E53    10-?01#   13-6
$RMEC2    9-374     9-;25     9-E57     9-K90     9-L32    10-?02#   10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11    10-?11
         13-6
$RMER1    9-325     9-328     9-331     9-334     9-337     9-340     9-343     9-346     9-349     9-354     9-357     9-398     9-415     9-417
          9-457     9-486     9-499     9-501     9-556     9-593     9-850     9-=00     9-=03    10->89#   13-6
$RMER2    9-360     9-362     9-852    10-?00#   13-6
$RMHR    10->98#   13-8
$RMLA    10->91#   13-7
$RMMR1   10->93#   13-7
$RMMR2   10->99#   13-6
$RMOF    10->96#   13-8
$RMSN    10->95#   13-8
$RMVEC    6-0#      8-56*     8-61      8-68     16-111
$RMWC     9-865     9-893     9-:92     9-;77     9-C49     9-D05     9-E24    10->84#   13-7
$RTNAD    9-P41#
$SAVR6    9-T76     9-T76#    9-T76*    9-T76*    9-T76*
$SAVRE    9-U09#    9-U15     9-U15
$SB2D     9-;18     9-F93     9-H38     9-H73     9-H73     9-H73     9-H73     9-H78     9-I11     9-I16     9-I21     9-S26#
$SB20     9-F77     9-S43#
$SEC      9-897     9-<66*    9->68     9-@53     9-A01     9-A29*    9-A76*    9-A84*    9-D24     9-L79*    9-M48*    9-M59*    9-M60     9-M78
          9-M81*   10-800    10->26#
$SETUP    4-266     4-266     4-266     4-266     4-266     4-266#    4-266#    4-266#    4-266#    4-266#    4-266#    8-13      8-13      8-13
          8-13      8-13      8-13      8-13      8-13      8-13      8-13      8-13      8-13      8-14      8-14      8-14      8-51      9-P41
          9-P41     9-T68     9-T68     9-T68     9-T68     9-U05     9-U05
$SIZE     8-92     16-85#
$SKI      9-F55     9-H77     9-I01     9-I01*   10->48#
$SOFT     9-H77     9-H99     9-H99*   10->46#
$SSEC     9-903     9-908     9-909     9-:95     9-;79     9-A34*    9-A37*    9-B06*    9-B10*    9-E27     9-L44*    9-L47*   10->35#
$STUP     4-266     4-266     4-266     4-266     4-266     4-266#    4-266#    4-266#    4-266#    4-266#    4-266#    4-266#    4-266#    4-266#
          4-266#
$SUPRS    9-;19     9-F18     9-F26     9-F31     9-F45     9-F50     9-F94     9-Q67#
$SVPC     4-272     4-272#
$SWR      4-52#     4-63      4-64      4-64      4-64      4-64      4-64      4-64      4-64      4-64      5-0       5-0       5-0       8-7
          8-13      9-P41     9-P41     9-P41     9-P41     9-P41     9-T68     9-T68     9-T68     9-T68     9-T68     9-T68     9-T68     9-T68
          9-T68     9-T68     9-T68     9-T76
$SWREG    5-0#      8-13
$TATUS    9-153     9-209     9-226     9-232     9-234     9-236     9-238     9-240     9-242     9-316     9-407     9-410     9-427     9-490
          9-<38     9-<74     9-<90     9-<96     9-=22     9-C46     9-D98     9-M56    10->30#   13-8
$TERM     9-U17#
$TESTN    5-0#      8-7*      9-P42*
$TIME     9-B80     9-H19     9-I07#    9-I68     9-T68
$TKB      5-0#      9-I69     9-J24     9-S57     9-T72     9-T72     9-U05     9-U05     9-U05     9-U05     9-U05     9-U05
$TKINT    8-50      8-207     9-S55#
$TKS      5-0#      9-J25*    9-S58*    9-S72*    9-S82*    9-S84*    9-T72     9-T72     9-U05     9-U05     9-U05     9-U05     9-U05     9-U05
          9-U05*
$TKSRV    9-S55     9-S65#
$TMPO     5-0#      8-125*    8-126     8-131
$TN       4-53#     4-63      8-7       8-7       8-7       8-7#
$TNPWR    9-U11     9-U11     9-U11#
$TOTAL    9-F20     9-H74     9-I03     9-I03*    9-063    10->45#
$TPB      5-0#      9-T72     9-T72     9-T72*
$TPFLG    5-0#      9-T72     9-T72     9-T72
$TPS      5-0#      9-T72     9-T72     9-T72
```

```
$TRANS   9-=28*   9-=29*   9-=34*   9-=35*   9-F23   10->43#
```

```
$TRAP    8-13      9-U15#
$TRAP2   9-U15     9-U15#
$TRK     9-<65*    9-@49     9-A81     9-A83*    9-D19     9-L80*    9-M47*    10->27#
$TRP     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15
         9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15
         9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15
         9-U15     9-U15     9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#    9-U15#
         9-U16     9-U16     9-U16     9-U16     9-U16#
$TRPAD   9-U15     9-U15#    9-U17
$TSTM    4-275#
$TSTNM   5-0#      9-T68     9-T68     9-T68
$TTYIN   9-S99     9-T00     9-T12     9-T35     9-T49     9-T53#
$TYPBN   9-U15
$TYPDS   9-U03#    9-U15     9-U15
$TYPE    9-Q86     9-Q99     9-T72#    9-T74     9-U15     9-U15
$TYPEC   9-T72     9-T72     9-T72     9-T72#    9-U05
$TYPEX   9-T72     9-T72     9-T72#
$TYPOC   9-U01#    9-U15     9-U15
$TYPON   9-U01     9-U01#    9-U15
$TYPOS   9-U01#    9-U15
$UNIT    5-0#      6-0
$UNITM   4-275#
$USWR    5-0#
$VECT1   5-0#      8-54      8-56
$VECT2   5-0#
$WRDL    9-867     9-892     9-:93     9-;76     9-=34     9-=60     9-=67     9-=69*    9-=70     9-=71*    9->02     9->11     9->16     9->20
         9->62     9-A31*    9-A95*    9-A96*    9-B02*    9-C48     9-E04     9-E25     9-L41*    10->34#   10->35    10->36    10->37    10->38
         10->39    10->40    10->41    10->42    10->43    10->44    10->45    10->46    10->47    10->48    10->49    10->50    10->51    10->55
$WRDM    9-<70*    9-A32*    9-A33*    9-A97*    9-A98*    9-A99     9-B01*    9-B03*    9-L42*    9-L43*    9-M49*    10->24#
$XOFF    9-T72     9-T72
$XON     9-T72     9-T72
.$ASTA   9-T74     9-T74
.$X      4-275     4-275#
A16      4-89#
A17      4-90#
ABASE    4-263#    5-0       5-0
ABNRML   9-161     9-061#
ACDW1    5-0       5-0
ACDW2    5-0       5-0
ACK      4-251#
ACPUOP   5-0       5-0
ACTDRV   10-92#    10-361*   10-412*   10-738*   10-747*   10-:09
ACTSTR   10-98#    10-:11*   10-:25*
ADDW0    5-0
ADDW1    5-0
ADDW10   5-0
ADDW11   5-0
ADDW12   5-0
ADDW13   5-0
ADDW14   5-0
ADDW15   5-0
ADDW2    5-0
ADDW3    5-0
ADDW4    5-0
ADDW5    5-0
ADDW6    5-0
```

```
ADDW8    5-0
ADDW9    5-0
ADEVCT   5-0       5-0
ADEVM    5-0       5-0
AENV     5-0       5-0
AENVM    5-0       5-0
AFATAL   5-0       5-0
AMADR1   5-0       5-0
AMADR2   5-0       5-0
AMADR3   5-0       5-0
AMADR4   5-0       5-0
AMAMS1   5-0       5-0
AMAMS2   5-0       5-0
AMAMS3   5-0       5-0
AMAMS4   5-0       5-0
AMSGAD   5-0       5-0
AMSGLG   5-0       5-0
AMSGTY   5-0       5-0
AMTYP1   5-0       5-0
AMTYP2   5-0       5-0
AMTYP3   5-0       5-0
AMTYP4   5-0       5-0
AOE      4-161#
APASS    5-0       5-0
APRIOR   5-0
APTCSU   9-T72     9-T74#
APTENV   9-T68     9-T72     9-T74     9-T74#
APTSIZ   8-13      9-T74#
APTSPO   9-T72     9-T74     9-T74#
ASGN1    9-J33#
ASGN2    8-205     9-J32     9-J48#
ASGN3    9-J44     9-J59     9-J67#
ASGN4    9-J68     9-K00#
ASGN6    9-J80     9-K02#
ASGN7    9-J79     9-K05#
ASGND    9-61     15-29#
ASKPAR   8-123    16-17#
ASNERR   9-J46     9-J64     9-K03     9-K16     9-K42     9-K61     9-027#
ASNLST   6-0#      9-18      9-68*     9-176     9-G79     9-G86     9-J39*    9-J42     9-J54*    9-J57     9-J67     9-K31     9-K33*    9-K53
         9-K63     9-K77     9-042*    9-051     9-P41     9-P41*
ASNMSG   9-J34*    9-J40*    9-J49*    9-J55*    9-K02*    9-K08*    9-K10*    9-K12*    9-K41*    9-K60*    9-032#
ASSIGN   9-J31#    9-K21     9-K98     9-L04     9-L09
ASWREG   5-0       5-0
AT0      4-174#
AT1      4-175#
AT2      4-176#
AT3      4-177#
AT4      4-178#
AT5      4-179#
AT6      4-180#
AT7      4-181#
ATA      4-148#
ATABIT   9-68      9-G86     9-J39     9-J42     9-J54     9-J57     9-J67     9-K31     9-K33     9-K53     9-K77     9-042     9-P41     10-152#
        10-327    10-424    10-530    10-928    10-950    10-958    10-959    10-979    10-:69
ATESTN   5-0       5-0
ATTN     6-0#      9-T68*   13-1      13-2
```

```
AUSWR    5-0       5-0
AUTOCK   6-0#
AVAIL    6-0#      9-22      9-62      9-66*     9-104     9-107     9-140     9-164
AVECT1   4-264#    5-0       5-0
AVECT2   5-0       5-0
BADENT   9-N42     9-N58     9-N73     9-010     15-45#
BADSEC   6-0#      9-160*    9-B57*    9-C26     9-H99     9-I00     9-I01     9-I02     9-I03
BAI      4-107#
BEGCOD   6-0#      9-L36     9-L37
BEGPAT   6-0#      9-L39
BEGSIZ   6-0#      9-L41     9-L42
BIT0     4-81#
BIT00    4-81      4-81#     9-834     9->63     9-?87
BIT01    4-81      4-81#     9-240     9-=30     10-404    10-659
BIT02    4-81      4-81#     9-883     9->59     9-@15     9-@22
BIT03    4-81      4-81#     9-316     9-684     10-886    10-;85
BIT04    4-81      4-81#     9-316     9-328     10-921
BIT05    4-81      4-81#     9-786     10-901
BIT06    4-81      4-81#     9-397     9-398     9-486     9-501     9-=22     9-D98     9-J25     9-S58     10-337    10-395    10-469    10-782
         10-<45
BIT07    4-81      4-81#     9-226     9-331     9-=22     10-337    10-604    10-759    10-886    10-901    10-921    10-954    10-<17
BIT08    4-81      4-81#     9-325     10-337
BIT09    4-81      4-81#     9-238     10-:60
BIT1     4-81#
BIT10    4-81#     9-236     9-343     9-351     9-T68     10-661
BIT11    4-81#     9-234     9-346     10-290    10-431    10-625    10-995
BIT12    4-81#     9-232     9-354     9-397     9-398     9-417     9-895     9->66     9-D86     10-284    10-318    10-337    10-406    10-439
         10-619    10-642    10-657    10-671    10-911    10-913    10-;25    10-;80    10-<46
BIT13    4-81#     9-334     9-360     9-656     9-752     9-Q96     9-T68     10-389    10-860    10-;28
BIT14    4-81#     9-213     9-215     9-240     9-319     9-322     9-360     9-477     9-527     9-573     9-614     9-720     9-735     9-912
         9-984     10-401    10-436    10-847    10-990    10-:90    10-;35    10-;39    10-<43
BIT15    4-81#     9-211     9-362     9-397     9-457     9-499     9-752     9-912     9-984     10-389    10-401    10-404    10-406    10-436
         10-439    10-625    10-659    10-661    10-782    10-886    10-901    10-911    10-921    10-990    10-:60    10-:90    10-:97
BIT2     4-81#     9-242     10-:97
BIT3     4-81#     9-337
BIT4     4-81#     9-579
BIT5     4-81#     9-340
BIT6     4-81#     9-417     9-C46
BIT7     4-81#     9-410     9-620     9-<96
BIT8     4-81#     9-532     9-556     9-562     9-593     9-598     9-752
BIT9     4-81#     9-349     9-752
BLKADR   6-0#      8-75      8-230     9-G84     9-J75     9-J94     9-K35     9-K56     9-T96
BLNKS1   9-B81     9-C32     9-H65     9-H73     9-H73     9-H73     9-H73     14-58#
BLNKS2   8-181     9-831     9-831     9-831     9-831     9-:41     9-:44     9-;57     9-;57     9-;57     9-;65     9-;65     9-;65     9-;97
         9-C35     9-C65     9-D16     9-D21     9-D37     9-E44     9-E44     9-E55     9-H36     9-H41     9-H49     9-H49     9-H57     14-57#
         16-106    16-115
BLNKS3   14-56#
BLNKS4   8-154     9-K80     14-55#
BPTVEC   4-81#
BUFTBL   6-0#      8-94*     8-95*     8-96*     8-99*     8-100*    8-102*    8-103*    8-106*    8-110     8-112     8-113     9-=57     9-=59
         9-=73*    9-=99     9->00     9->12*    9->17*    9->22     9->23     9->36*
BUSADR   8-66      16-98#
BUSY     9-I64     15-46#
CFLAG    6-0#      8-85*     9-3       9-I67*    9-I74     9-M08*    9-M13     9-N09*    9-N31     9-N55     9-N70     9-N93*    9-005     9-015*
         9-S71*    9-T28*
CHGADR   6-0#      8-3*      8-6*      8-203     8-218     9-T93*    16-98     16-100*
```

```
CHKADP   9-198     9-?92     9-@40#
```

```
CI1      10-459    10-487#
CI3      10-461    10-510#
CI4      10-450    10-532#
CI5      10-509    10-531    10-609#   10-618
CI6      10-548    10-558    10-560    10-562    10-614#
CI7      10-391    10-468    10-499    10-503    10-507    10-515    10-525    10-529    10-540    10-547    10-553    10-557    10-571    10-577
         10-581    10-591    10-602    10-617    10-619#   10-733    10-763    10-940    10-<29
CI7B     10-627#   10-944
CI8      10-630    10-641#   10-813    10-:82
CK.CHR    9-R52#    9-R89     9-R97
CK.DEC    9-R30#    9-R58
CK.DIG    9-N41     9-N57     9-N72     9-007     9-R83#
CK.NUM   16-109    16-118    16-138#
CK.OCT    9-R12#    9-R60
CKBUS     9-218     9-865#
CKCLK     8-143     9-G40#    9-T87
CKCLK1    9-G42     9-G53#
CKCLK2    9-G54     9-G62#
CKCLK3    9-G52     9-G61     9-G69#
CKERR     9-217     9-846#
CKFMT     9-330     9-556#
CKHCE     9-333     9-593#
CKSWR     9-T68     9-T68     9-U15#
CLKFLG    6-0#      9-G40*    9-G45*    9-G56*    9-I07
CLOCK     9-G48     9-G58     9-I28#
CLR       4-109#   10-242    10-680
CLRDPB    8-231     9-J84     9-L19#
CLRQUE   10-224    10-679    10-=13#
CMCNT     6-0#      9-833*    9-837*    9-892*    9-893*    9-903     9-905     9-906*    9-909*    9-960     9-964     9-967*    9-991*
CMCYL     6-0#      9-894*    9-895*    9-896*    9-913     9-980*    9-981     9-985
CMDAT     9-911     9-921     9-926#    9-968     9-970
CMHED     9-912#
CMPAR     9-221     9-883#
CMPARD    9-446     9-887#
CMPLMT    6-0#      9-841     9-898
CMPRES    9-42      9-96      9-142     9-168     9-A48#    9-A51
CMPRT     9-923     9-938     9-948     9-958     9-:00#
CMPRX     9-920     9-961     9-966     9-983     9-992     9-995#
CMSEC     6-0#      9-897*    9-916     9-971*    9-972     9-974*    9-988
CMSTR     9-900#
CMSTR2    9-915     9-918     9-922#    9-987     9-990
CMTRK     6-0#      9-975*    9-977     9-979*
COLON     9-I14     9-I19    15-40#
COMTBL    6-0#      9-A14     9-L38
CR        4-81#     9-T72     9-T72    12-7      12-9      12-10     14-42     14-43     14-44     14-46     14-48     14-49     14-51     14-52
         14-53     14-54
CRLF      4-81#     8-14      8-14      8-28      8-39      8-45      8-46      9-T54     9-T72     9-T72    15-14     15-18     15-19     15-21
         15-23     15-25     15-27     15-28     15-30     15-30     15-34     15-35     15-36     15-37     15-38     15-38     15-39     15-39
         15-41     15-42     15-43     15-43     15-44     15-45     15-46     15-47     15-47     15-49     15-50     15-51     16-17     16-81
CYLIMT    6-0#      8-241     9-L66     9-N41
CYLNDR    9-600*    9-601     9-604*    9-638*    9-D85     9-E44     9-E44     9-M52    10-?20    16-171#
DATA0     6-0       6-0#
DATA1     6-0       6-0#
DATA10    6-0       6-0#
DATA11    6-0       6-0#
DATA12    6-0       6-0#
```

```
DATA14   6-0        6-0#
DATA15   6-0        6-0#
DATA2    6-0        6-0#
DATA3    6-0        6-0#
DATA4    6-0        6-0#
DATA5    6-0        6-0#
DATA6    6-0        6-0#
DATA7    6-0        6-0#
DATA8    6-0        6-0#
DATA9    6-0        6-0#
DATAPK   9-J02      9-K97#
DATE     6-0#       9-K67      9-K70      16-66*     16-69*     16-69*     16-69*     16-69*     16-69*     16-69*     16-69*     16-70*
DATEIS   9-K69      15-41#
DCK      4-167#
DCKER    9-359      9-370#
DCKER1   9-393#     9-674
DDISP    4-81#      5-0        8-13
DEASGN   9-I92      9-K25#
DEASSG   9-38       15-27#
DH1      7-5        12-1#
DH14     9-C31      12-6#
DH15     9-C40      12-8#
DH16     9-C43      12-10#
DH2      7-12       7-33       12-2#
DH3      7-19       12-3#
DH4      7-26       12-4#
DH6      7-40       12-5#
DISPLA   5-0#       8-13*      8-13*      9-T68*
DISPLY   9-250      9-283      9-295      9-306      9-378      9-392      9-413      9-450      9-460      9-481      9-522      9-547      9-569      9-610
         9-635      9-651      9-673      9-679      9-697      9-707      9-716      9-731      9-747      9-767      9-779      9-798      9-816      9-821
         9-822      9-823      9-831      9-831      9-831      9-831      9-831      9-831      9-831      9-831      9-855      9-873      9-:21      9-:26
         9-:27      9-:28      9-:41      9-:44      9-:47      9-:56      9-:59      9-;15      9-;20      9-;52      9-;57      9-;57      9-;57      9-;60
         9-;65      9-;65      9-;65      9-;67      9-;68      9-;91      9-;97      9-<02      9-<04      9-=13      9-B76      9-B77      9-B79      9-B81
         9-B95      9-C21      9-C23      9-C28      9-C29      9-C30      9-C31      9-C32      9-C35      9-C40      9-C43      9-C54      9-C55      9-C65
         9-C68      9-C74      9-C80      9-C87      9-D01      9-D04      9-D07      9-D13      9-D16      9-D17      9-D21      9-D22      9-D26      9-D34
         9-D37      9-D38      9-D41      9-D51      9-D56      9-D58      9-D61      9-D65      9-D69      9-D74      9-D77      9-D84      9-D88      9-D92
         9-E00      9-E03      9-E06      9-E11      9-E17      9-E21      9-E30      9-E32      9-E39      9-E44      9-E44      9-E45      9-E46      9-E52
         9-E55      9-E56      9-E59      9-E65      9-E66      9-E72      9-E78      9-E84      9-E90      9-E97      9-E99      9-F07      9-F08      9-F14
         9-F19      9-F22      9-F27      9-F32      9-F41      9-F46      9-F51      9-F54      9-F57      9-F66      9-F80      9-Q75      9-U16#
DISPRE   4-268#     8-13
DLT      4-119#
DONE     9-228      9-316#
DPE      4-232#
DPINT    10-60#     10-253     10-256     10-345*    10-371     10-831     10-856     10-980     10-982*    10-:51     10-:71     10-:76     10-:84*
DPR      4-141#
DPRQS    10-69#     10-377     10-425*    10-463*    10-632*    10-649     10-665*    10-834     10-:53     10-:73     10-:78     10-:93*
DRIVE    6-0#       9-T68*     13-2       13-3       13-4
DRIVE0   6-0        9-K76      10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE1   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE2   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE3   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE4   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE5   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE6   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRIVE7   6-0        10-?11#    16-54      16-54      16-54      16-54      16-54      16-54
DRNUM    9-J85      15-28#
```

```
DROPNG   9-046     15-22#
DRQ      4-197#
DRVACT   10-29#    10-385    10-565*   10-612*   10-634*   10-646    10-664*   10-752*   10-852    10-884    10-915*   10-922*   10-937    10-949*
         10-:63*
DRVCLR   4-246#
DRVER    9-364     9-546#
DRVINT   10-244    10-278#   10-373    10-970    10-983
DRVPRM   9-J90     9-L58#
DRVQUE   10-380    10-393    10-795    10-=53#
DRVSTA   8-155     8-228     9-J78     10-39#    10-234*   10-235*   10-236*   10-237*   10-247*   10-279*   10-289*   10-333*   10-339*   10-367
         10-375    10-399    10-433    10-437    10-566*   10-683*   10-837    10-845    10-858    10-909*   10-916*   10-929    10-985    10-:85*
DRVTYP   8-159     8-183     8-186     9-K05     9-L96     10-51#    10-280*   10-296*   10-301*   10-306*   10-311*   10-402    10-684*
DRY      4-140#
DSWR     4-81#     5-0       8-13
DT00     4-188#
DT01     4-189#
DT02     4-190#
DT03     4-191#
DT04     4-192#
DT05     4-193#
DT06     4-194#
DT07     4-195#
DT08     4-196#
DT1      7-6       13-1#
DT14     9-C36     13-6#
DT15     9-C41     13-7#
DT16     9-C44     13-8#
DT2      7-13      7-34      13-2#
DT3      7-20      13-3#
DT4      7-27      13-4#
DT6      7-41      13-5#
DTE      4-164#
DTEER    9-356     9-670#
DTUW     10-146#   10-230    10-452    10-508*   10-639*   10-652    10-666    10-668*   10-677*   10-741    10-753*   10-:56    10-:66*   10-;33
DULP     10-279#   10-291
DUNIT    6-0#      9-12      9-K35*    9-044*    9-P41*
DVA      4-134#
DVC      4-228#
ECBAD0   6-0#      9-;33*    9-;57
ECBAD1   6-0#      9-;47*    9-;65
ECBIT    6-0#      9-;07*    9-;08*    9-;09     9-;10*    9-;27     9-;29*
ECC      9-439     9-:91#
ECC1     9-;40     9-;46     9-;52#
ECC2     9-;24     9-;67#
ECCX     9-;59     9-;66     9-;68#
ECGD     6-0#
ECGD1    6-0#
ECH      4-158#
ECI      4-216#
ECMSK0   6-0#      9-;25*    9-;30*    9-;35     9-;37
ECMSK1   6-0#      9-;26*    9-;31*    9-;39     9-;48     9-;50
ECSEC    6-0#      9-:91*    9-;00*    9-;03*    9-;05*    9-;21
ECWRD    6-0#      9-;09*    9-;11*    9-;12*    9-;13*    9-;14*    9-;16     9-;21*    9-;22     9-;33     9-;36     9-;37*    9-;38*    9-;41
         9-;57     9-;57
ECWRD1   6-0#      9-;34*    9-;41*    9-;42*    9-;43     9-;45*    9-;47     9-;49     9-;50*    9-;51*    9-;58     9-;65     9-;65
EM1      7-4       11-3#
```

```
EM11      9-269    11-10#
EM12      9-250    11-11#
EM13      9-283    11-12#
EM14      9-295    11-13#
EM15      9-306    11-14#
EM2       7-11     11-4#
EM20      9-522    11-15#
EM21      9-392    11-16#
EM22      9-476    11-17#
EM23      9-460    11-18#
EM24      9-569    11-19#
EM25      9-610    11-20#
EM26      9-716    11-21#
EM27      9-731    11-22#
EM3       7-18     11-5#
EM30      9-547    11-23#
EM31      9-651    11-24#
EM32      9-673    11-25#
EM33      9-679    11-26#
EM34      9-779    11-27#
EM35      9-697    11-28#
EM36      9-707    11-29#
EM37      9-479    11-30#
EM4       7-25     11-6#
EM40      9-747    11-31#
EM41      9-873    11-32#
EM42      9-:21    11-33#
EM43      9-816     9-821    11-34#
EM44      9-855    11-35#
EM45      9-378    11-36#
EM46      9-C54    11-37#
EM5       7-32     11-7#
EM50      9-767    11-38#
EM51      9-635    11-39#
EM6       7-39     11-8#
EM60      9-798    11-40#
EMPTYQ   10-631    10-682    10-783    10-:98    10-=38#
EMTVEC    4-81#     8-13*     8-13*
ENDCMP    9-995     9-:52#
ENDCON    6-0#      8-210*    8-211*    8-215*    8-216*    9-P41     9-P41
ENDET     6-0#      9-P41    16-9
ENDPAS    9-P41    15-23#
ENDPGM    8-95      8-100    16-173#
ENDSEK    6-0#      9-P41     9-P41
ENDTST    9-P41    15-25#
ENTADR    9-N10    15-39#
ENTCOM    9-I70    15-36#
ENTDAT   16-63     16-81#
ENTDRV    9-M09    15-37#
ENTID    16-71     16-82#
ENTLMT    9-L62    15-38#
ENTPR     8-134#
EOP1      9-P41     9-P41#
EOP2      9-A89     9-P41     9-P41     9-P41     9-P41#
EOPX      9-P41     9-P41     9-P41     9-P41     9-P41     9-P41     9-P41#
ERCTR     6-0#      9-890*    9-922*    9-934     9-947*    9-:54     9-:57
```

```
ERPROC   9-210     9-214     9-216     9-226#
ERR      4-147#
ERROR    4-81#
ERRVEC   4-81#     8-13      8-13*     8-13*     9-G42*    9-G43*    9-G54*    9-G69*   16-85     16-85     16-85*    16-85*    16-85*    16-85*
F0       4-129#
F1       4-130#
F2       4-131#
F3       4-132#
F4       4-133#
FACTOR   9-=24*    9-=25*    9-=27*    9-=28     9-=36     9-=44#
FAIRNS   6-0#
FALPAR   9-237     9-267#
FALPR1   9-263     9-271#
FER      4-156#
FILBUF   9-88      9-120     9->58#
FISK     9-P41     9-P42#
FMT16    4-217#
FMTER    9-564     9-715#
FORMAT   6-0#      9-188     9-@65    16-10
FOWT1    8-210#
FOWT2    8-217     8-221#
FOWT3    8-219     8-257#
FRSTER   6-0#      9-445*    9-447     9-813     9-839*    9-840*    9-886*    9-950     9-:02     9-:17     9-:29*    9-:52
GENDPB   9-423*    9-427     9-<13*    9-<14*    9-<16     9-<31*    9-<32*    9-<36     9-<38     9-<49*    9-<50*    9-<52     9-<65*    9-<66*
         9-<67*    9-<68*    9-<69*    9-<70*    9-<72     9-<74     9-J73*    9-M45*    9-M46*    9-M47*    9-M48*    9-M49*    9-M50*    9-M54
         9-M56     9-M59*    9-M60     9-M78     9-M81*   10-?15#
GENREG   9-562     9-598    10-?24    10-?27#
GETADR   8-232     9-J91     9-M35#
GETBUF   9-84      9-117     9-=54#
GETID    9-J89     9-M05#
GETLMT   8-242     9-976     9-?54     9-G32     9-L68     9-L92#    9-M36
GETPAR   9-115     9-@95#
GETPAT   9-201     9-@26     9-@75#    9-@84     9-B13
GETREG   4-253#
GETREM   9-?16     9-?45     9-?63     9-?80     9-@03     9-@80     9-P51#
GETREQ  10-426    10-654    10-774    10-841    10-875    10-939    10-989    10-:59    10-:87    10-:94    10-=74#
GNS      4-268     4-268     8-14      8-28      8-39      8-45      8-46      9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15
         9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15     9-U15
         9-U15     9-U16     9-U16
GO       4-128#
GODRIV   9-89      9-122     9-406     9-489     9-<89     9->92#
GTSWR    8-14      8-51      9-U15#
HCE      4-159#
HCEER    9-605     9-730#
HCI      4-215#
HCRC     4-160#
HCRCER   9-327     9-517#    9-558     9-595
HEADER   6-0#      9-?25    16-13
HEDLIN   9-K75    15-43#
HOUR     6-0#      8-80*     9-I10     9-I40*    9-I41     9-I43*    9-T83*
HT       4-81#     9-T72     9-T72
HZ       6-0#      8-79      9-I30
IAE      4-162#
IAEER    9-345     9-696#
IDIS     9-K73    15-42#
IDLE     9-7       9-102     9-106     9-150#
```

```
IE        4-87#
ILF       4-152#
ILR       4-153#
INCHRD    9-431     9-I00#
INCMIS    9-640     9-I02#
INCSKI    9-771     9-I01#
INCSOF    9-438     9-444     9-530     9-577     9-618     9-H99#
INCTOT    9-254     9-274     9-287     9-299     9-310     9-380     9-412     9-432     9-449     9-465     9-507     9-510     9-531     9-550
          9-578     9-619     9-641     9-655     9-683     9-700     9-709     9-724     9-739     9-751     9-770     9-783     9-801     9-859
          9-877     9-:60     9-I03#
INTDON    8-257    15-47#
INTRVL    6-0#      8-83*     9-I35*    9-I46     9-I48     9-I48     9-I51*    9-T86*    9-T88     9-T90*   16-4
INVLD     9-J21    15-35#
IOTVEC    4-81#
IR        4-110#
ISR      10-239    10-738#
IVC       4-231#
KIPAR0   16-85
KSR       9-8       9-I62#
KSR1      9-I63     9-I65#
LA       10-456    10-699#
LACNT    10-113#   10-707*   10-708    10-729*
LBC       4-229#
LBT       4-143#
LF        4-81#     9-T72     9-T72    12-7      12-9      12-10     14-42     14-43     14-44     14-46     14-48     14-49     14-51     14-52
         14-53     14-54     15-18     15-43
LIMIT     6-0#      9-841*    9-898*    9-899*    9-:31     9-:33*
LIN10A    9-;15    14-47#
LIN10B    9-;20    14-48#
LIN10C    9-;67    14-49#
LIN10H    9-;52    14-51#
LIN11H    9-;91    14-53#
LIN2C     9-B98    14-1#
LIN2P     9-C04    14-2#
LIN2S     9-C29    14-3#
LIN3.1    9-C75     9-C81     9-D49#
LIN3.3    9-C86     9-C93     9-D74#
LIN3.4    9-C94     9-D84#
LIN6.1    9-E73     9-E79     9-E95#
LIN6.2    9-E85     9-E91     9-F04#
LIN7M     9-F51    14-33#
LIN7O     9-F14     9-F41    14-34#
LIN7P     9-F46    14-35#
LIN7R     9-F27    14-39#
LIN7S     9-F54    14-36#
LIN7T     9-F19    14-37#
LIN7X     9-F22    14-38#
LIN8M     9-413     9-=13     9-F66    14-40#
LIN9B     9-:26    14-41#
LIN9E     9-:56    14-45#
LIN9G     9-450    14-46#
LIN9H     9-:28    14-42#
LIN9I     9-823    14-44#
LINA3     9-D84    14-11#
LINB3     9-D01    14-15#
LINB5     9-E21    14-23#
```

```
LINC6     9-E72     14-29#
LINCA3    9-D38     14-13#
LIND5     9-E17     14-22#
LINDA3    9-D34     14-14#
LINDEC    9-:58     9-C34     9-D15     9-D20     9-D25     9-D50     9-D55     9-D57     9-D60     9-D64     9-D68     9-D76     9-D79     9-D87
          9-D91     9-E05     9-E10     9-E31     9-F06     9-F21     9-F53     9-F56     9-F92#
LINE1     9-249     9-282     9-294     9-305     9-377     9-391     9-459     9-475     9-521     9-546     9-568     9-609     9-634     9-650
          9-672     9-678     9-696     9-706     9-715     9-730     9-746     9-766     9-778     9-797     9-815     9-854     9-872     9-:20
          9-B71#
LINE2     9-251     9-284     9-296     9-309     9-379     9-393     9-464     9-485     9-523     9-548     9-570     9-611     9-636     9-652
          9-680     9-698     9-708     9-717     9-732     9-748     9-768     9-780     9-799     9-817     9-856     9-874     9-:22     9-B94#
          9-F67
LINE2A    9-C03     9-C08     9-C26#
LINE2B    9-C27     9-C30#
LINE3     9-252     9-285     9-297     9-309     9-394     9-464     9-485     9-524     9-549     9-571     9-612     9-653     9-718     9-733
          9-749     9-800     9-857     9-C74#
LINE3A    9-781     9-818     9-:23     9-C80#
LINE3B    9-769     9-C86#
LINE3C    9-637     9-C93#
LINE3D    9-875     9-C56     9-C99#
LINE3E    9-681     9-D13#
LINE3F    9-699     9-D32#
LINE4     9-253     9-286     9-298     9-309     9-395     9-464     9-485     9-525     9-572     9-613     9-654     9-682     9-719     9-734
          9-750     9-782     9-819     9-858     9-876     9-:24     9-C57     9-D98#
LINE5     9-464     9-485     9-529     9-575     9-616     9-722     9-737     9-E17#
LINE5A    9-526     9-576     9-617     9-639     9-723     9-738     9-E38#
LINE5B    9-437     9-E52#
LINE6     9-400     9-E65#
LINE6A    9-443     9-E72#
LINE6B    9-436     9-E78#
LINE6C    9-384     9-469     9-494     9-503     9-536     9-583     9-624     9-660     9-688     9-758     9-789     9-805     9-E84#
LINE6D    9-386     9-441     9-471     9-539     9-586     9-627     9-663     9-691     9-760     9-792     9-807     9-E90#
LINE7     9-255     9-288     9-300     9-311     9-387     9-433     9-451     9-508     9-511     9-537     9-540     9-551     9-584     9-587
          9-625     9-628     9-661     9-664     9-689     9-701     9-710     9-725     9-740     9-761     9-790     9-808     9-860     9-878
          9-:61     9-=10     9-F14#
LINE7A    9-642     9-772     9-F41#
LINE8     9-430     9-505     9-=09     9-F66#
LINEN3    9-D77     14-10#
LINEO5    9-E56     14-27#
LINEP5    9-E52     14-26#
LING6     9-E84     14-30#
LINKDV    9-:96     9-;80     9-E28     9-P54     9-P64#
LINM3     9-C74     14-4#
LINM4     9-E00     14-19#
LINN3     9-C80     14-6#
LINOCT    9-831     9-831     9-831     9-831     9-831     9-831     9-831     9-831     9-:40     9-:43     9-:46     9-;57     9-;57     9-;57
          9-;65     9-;65     9-;65     9-;94     9-;99     9-C63     9-D03     9-D06     9-D36     9-D40     9-E02     9-E20     9-E23     9-E44
          9-E44     9-E54     9-E58     9-F76#
LINP3     9-D58     14-8#
LINP5     9-E30     14-24#
LINR6     9-F07     14-31#
LINS3     9-D13     9-D74     14-9#
LINS4     9-E03     14-20#
LINS5     9-E39     14-25#
LINSS3    9-D22     14-18#
LINST3    9-D17     14-17#
```

```
LINU06   9-E90    14-32#
LINW3    9-D04    14-16#
LINX4    9-E06    14-21#
LINX5    9-E45    14-59#
LKPAR    8-117#
LODEV    8-178    9-J40    9-J55    15-13#
LSC      4-230#
LSTAD    6-0#     8-93*    8-96     8-97
M.DP40   9-P97#   9-Q18
M.DP41   9-Q02    9-Q09#
M.DP42   9-Q08    9-Q13#
M.DP44   9-Q20    9-Q25#
M.DP50   9-P95    9-Q29#
M.DPID   9-P71    9-P86#
MAIN     9-3#     9-180    9-A94    9-B20
MAIN1    8-255    8-258    9-19     9-47#    9-P43
MAIN2    9-57     9-75#
MAIN3    9-81     9-101#
MAINDA   9-11#
MANTER   9-J93    9-N05#
MASK     6-0#     9-397*   9-415    9-461*   9-532*   9-579*   9-620*   9-656*   9-684*   9-755*   9-786*   9-<98    9-=03
MATCH    9-926    9-:72#
MAXCYL   8-241*   9-?29    9-?73    9-A78    9-L66*   10->65#  10->66   10->67   10->68   10->69   10->70   10->71   10-->75  16-54
         16-54    16-54    16-54    16-54    16-54    16-54    16-54
MAXDL    6-0#     8-107    8-109*   8-110    8-112*   8-136    8-138*   9-?99    9-@12    9-@14    9-A95    9-A97    16-3
MAXER    6-0#     8-86*    9-063
MAXSEC   8-244*   9-?38    9-L71*   10->69#  16-54    16-54    16-54    16-54    16-54    16-54    16-54    16-54
MAXTRK   8-243*   9-?55    9-A81    9-L69*   10->67#  16-54    16-54    16-54    16-54    16-54    16-54    16-54    16-54
MCPE     4-92#
MCPEMX   10-164#  10-;15   10-;61
MDPE     4-112#
MERR1    9-M86    15-49#
MERR2    9-M84    15-50#
MINCYL   8-245*   9-?31    9-?74    9-?76    9-?81    9-A85    9-L67*   9-L81    10->66#  16-54    16-54    16-54    16-54    16-54
         16-54    16-54    16-54
MINSEC   8-247*   9-?32    9-?39    9-?41    9-?46    9-A84    9-L32    9-L72*   9-L79    10->70#  16-54    16-54    16-54    16-54
         16-54    16-54    16-54    16-54
MINTRK   8-246*   9-?33    9-?56    9-?58    9-?64    9-A83    9-L70*   9-L80    10->68#  16-54    16-54    16-54    16-54    16-54
         16-54    16-54    16-54
MINUTE   6-0#     8-81*    9-I15    9-I36*   9-I37    9-I39*   9-T84*   9-T92*
MNDLTA   10-179#  10-727
MNTBL    6-0#     9-C19
MOH      4-198#
MOL      4-145#
MONTR    8-201#
MRMCS1   16-103   16-126#
MRMVEC   16-112   16-127#
MSFULL   9-N20    15-51#
MSGCYL   9-N26    15-52#
MSGFOR   9-P41    15-26#
MSGON    9-271    9-047    9-P41    15-24#
MSGSEC   9-N65    15-54#
MSGTRK   9-N50    15-53#
MSWRO    9-J19    15-34#
MXDLTA   10-176#  10-725
MXF      4-113#
```

```
MXWNDW   10-182#
N         8-128     15-31#
NED       4-116#
NEDCLK    9-G63     15-30#
NEM       4-115#
NEWASN    9-I88      9-K20#
NEWUNT    6-0#       8-234*     9-51       9-66       9-67*      9-J94*
NOMTCH    9-813#     9-928
NONE      9-K83     15-44#
NOTAVL   15-11#
NOTPRS    8-165      9-K10     15-10#
NOTPRT    6-0#       9-B55     16-12
NOTRM     8-162      9-K08     15-9#
NOTSAF    8-171      9-K02     15-12#
NSA       4-200#
OFFCOD    6-0#       9-423
OFFDIR    4-214#
OFFON     4-138#
OFFSET    4-248#     9-<50
OFFST     9-426      9-<49#
OFLIN     9-241      9-293#
OFMSG0    6-0        6-0#
OFMSG1    6-0        6-0#
OFMSG2    6-0        6-0#
OFMTBL    6-0#       9-E96
ONES      6-0        6-0#
OPE       4-227#
OPERID    6-0#       9-K71      9-K74     16-74*     16-77*     16-77*     16-77*     16-77*     16-77*     16-78*
OPI       4-165#
OPIER     9-336      9-648#
OPIER1    9-657#
OPRDAT    8-71      16-63#
OPT      10-387     10-422#    10-777     10-963     10-:01
OPTBL     6-0#       9-C10      9-C12
OR        4-111#
ORDERQ    6-0#       8-73       9-5        9-90       9-101      9-123      9-150      9-I62      9-T94
PACK      6-0#       8-84*      8-254*     9-B17*     9-J96      9-J98      9-K20*     9-K97*     9-L03*     9-L08*
PAR       4-155#
PAR1     16-3       16-18#
PAR10    16-10      16-27#
PAR11    16-7       16-28#
PAR14    16-8       16-29#
PAR15    16-30#
PAR16    16-9       16-31#
PAR19    16-5       16-32#
PAR2     16-4       16-19#
PAR20    16-12      16-33#
PAR21    16-13      16-34#
PAR3     16-6       16-20#
PAR4     16-21#     16-54      16-54      16-54      16-54      16-54      16-54      16-54      16-54
PAR5     16-22#     16-54      16-54      16-54      16-54      16-54      16-54      16-54      16-54
PAR6     16-23#     16-54      16-54      16-54      16-54      16-54      16-54      16-54      16-54
PAR7     16-24#     16-54      16-54      16-54      16-54      16-54      16-54      16-54      16-54
PAR8     16-25#     16-54      16-54      16-54      16-54      16-54      16-54      16-54      16-54
PAR9     16-26#     16-54      16-54      16-54      16-54      16-54      16-54      16-54      16-54
PARENT    8-135      9-L78      9-N92#
```

```
PARLST   8-113*    8-134     16-3#
PARQ     6-0#
PASCNT   6-0#      9-P41     16-5
PAT      4-108#
PATTEN   6-0#      9-@76     9-@78     16-6
PCLOCK   6-0#      9-G41*    9-G46*
PERIOD   15-33#
PGE      4-114#
PGM      4-142#
PIP      4-146#
PIRQ     4-81#
PIRQVE   4-81#
POPQUE   10-435    10-487    10-603    10-924    10-953    10-993    10-=89#
POSER    9-603     9-634#
PR0      4-81#
PR1      4-81#
PR2      4-81#
PR3      4-81#
PR4      4-81#     9-I66
PR5      4-81#
PR6      4-81#
PR7      4-81#
PROCES   9-159     9-208#
PRTBAD   9-396     9-434     9-784     9-;73#
PRTIM    9-243     9-305#
PS       4-81      4-81#     8-78*     9-I66*    9-J26*    9-Q83     9-Q84*    9-Q98*    9-S80     9-T82*   10-222    10-223*   10-252*   10-260*
         10-359    10-360*   10-413*   10-423    10-472*   10-714*
PSEL     4-91#
PSW      4-81#
PUNSAF   9-233     9-248#
PWRVEC   4-81#     8-13*     8-13*     9-T76*    9-T76*    9-T76*    9-T76*    9-T76*    9-T76*
QCNT     10-<55#   10-=14    10-=38*   10-=53    10-=55*   10-=75    10-=89*
QDRV0    10-<66    10-<77    10-<86    10-<98#
QDRV1    10-<67    10-<78    10-<87    10-<99#
QDRV2    10-<68    10-<79    10-<88    10-=00#
QDRV3    10-<69    10-<80    10-<89    10-=01#
QDRV4    10-<70    10-<81    10-<90    10-=02#
QDRV5    10-<71    10-<82    10-<91    10-=03#
QDRV6    10-<72    10-<83    10-<92    10-=04#
QDRV7    10-<73    10-<84    10-<93    10-=05#
QINPT    10-<66#   10-=40    10-=57*   10-=58*   10-=59    10-=61*
QOUTPT   10-<77#   10-=40*   10-=78    10-=91    10-=92*   10-=93*   10-=94    10-=96*
QSTART   10-<86#   10-=20    10-=25    10-=61    10-=96
QSTOP    10-<87#   10-=59    10-=94
QTERM    10-<94    10-=06#
QUES     9-028     15-3#
R6       4-81#     8-13      8-13*     8-13*
R7       4-81#
RANCYL   9-?73#
RANPAT   9-?94     9-@26#
RANSEC   9-?26     9-?38#
RANSIZ   9-?88     9-?99#    9-@07
RANTRK   9-?54#
RANWRT   9-?19     9-@64#
RANXIT   9-?91     9-@23     9-@28#
RATIO    6-0#      8-214*    9-?17     16-8
```

```
RD.RM    10-292    10-322    10-328    10-551    10-575    10-589    10-701    10-715    10-761    10-811    10-878    10-888    10-903    10-:45
         10-;15#   10-;68    10-;82    10-;88    10-<21
RD.RM1   10-;19#   10-;47
RD.RM2   10-;15*   10-;46#
RD.RM3   10-;26    10-;38    10-;48#
RD.RM4   10-;31    10-;50#
RD.WRD   10-;21#   10-;22    13-3      13-4
RDCHR     8-124     9-S65     9-T02     9-U15#
RDDAT     4-260#    6-0       9-M50
RDHD      4-261#    6-0       9-<69     9-B08
RDLIN     9-I72     9-M11     9-N29     9-N53     9-N68     9-003     9-U15#   16-64     16-72     16-107    16-116
RDY       4-88#
READDR    9-192     9-519     9-560     9-596     9-A03     9-A72     9-B33     9-D49     9-G15#
READHD    9-520     9-561     9-597     9-<65#
READIN    4-250#
RECAL     4-245#    9-<32
RECALO    9-<32#    9-J76
RECALT    9-643     9-773     9-<31#
REDAPK    9-J06     9-L08#
REFMT     9-184#    9-452     9-512     9-541     9-588     9-629     9-665     9-741
REFMTX    9-185     9-187     9-189     9-191     9-200     9-204#
RELBUF    9-169     9-=95#
RELSE     4-247#
REPHD     9-178    15-18#
REPLZ     9-H39     9-H49     9-H49     9-H57     9-H65     9-H73     9-H73     9-H73     9-H73     9-H79     9-I12     9-I17     9-I22     9-Q40#
RESREG    9->83     9-J23     9-P74     9-U11     9-U13     9-U15#   10-261    10-407    10-410    10-473    10-686    10-746    10-:24    10-<30
         10-=29
RESVEC    4-81#
RETRY     6-0#      9-381*    9-402*    9-419*    9-420     9-420     9-425*    9-466*    9-488*    9-493*    9-496*    9-497     9-497     9-533*
          9-580*    9-621*    9-657*    9-685*    9-754*    9-785*    9-802*    9-<93*    9-=05*    9-=06     9-=06     9-F05
RM05      9-<15     9-<35     9-<51     9-<71     9->94     9-M53    10-359#
RMADR     8-60*     8-67*    10-169#   10-241    10-366    10-490    10-510    10-532    10-699    10-740    10-:43    10-;17    10-;23    10-;27
         10-;41    10-;74    10-;78    10-<10    10-<40    16-121
RMAS     10-193#   10-327*   10-809    10-928*   10-958*   10-979*   10-:68
RMBA     10-188#
RMCS1    10-186#   10-283*   10-290    10-316    10-430*   10-431    10-506    10-528    10-570    10-590    10-601    10-616    10-629    10-762
         10-812    10-865    10-895    10-934    10-995    10-;69
RMCS2    10-190#   10-242*   10-282*   10-284    10-379*   10-429*   10-491*   10-511*   10-533*   10-619    10-642    10-657    10-671    10-680*
         10-700*   10-760*   10-876*   10-965*   10-978*   10-:44*   10-;24    10-;79    10-<11*   10-<17    10-<41*   10-<46
RMDA     10-189#   10-498    10-524    10-539
RMDB     10-195#   10-<15
RMDC     10-200#   10-502    10-514    10-546
RMDS     10-191#   10-323    10-702    10-879    10-904    10-966    10-:46
RMDT     10-197#   10-293
RMEC1    10-204#
RMEC2    10-205#   10-<25
RMER1     9-562     9-598    10-192#   10-329    10-889    10-967    10-;83
RMER2    10-203#   10-968
RMERRS   10-22#    10-225    10-847    10-850    10-860    10-966*   10-967*   10-968*   10-969*   13-2      13-2      13-2      13-2
RMHR     10-201#
RMINIT    8-144    10-221#
RMLA     10-194#   10-716    10-718
RMMR1    10-196#
RMMR2    10-202#   10-969
RMOF     10-199#   10-320    10-552    10-556    10-576    10-580
RMR       4-154#
```

```
RMTMR    9-I45   10-:09#
RMVEC    8-61*    8-68*   10-170#  10-238   10-240   10-360
RMWC    10-187#
RNOP     4-243#
RTC      4-249#   9-<14
RTNCTR   9-<13#
S        9-D56    9-D65   14-7#
SATPOW   9-T76    9-T81#
SAVEFG   8-145*  10-124#  10-605   10-766   10-955
SAVER1   6-0#     9-901*   9-953*   9-:00    9-:08    9-:12*
SAVER5   6-0#     9-902*   9-954*   9-:09    9-:13*
SAVREG   9->58    9-I65    9-P64    9-U11    9-U13    9-U15#  10-221   10-362   10-422   10-641   10-739   10-:12   10-<07   10--=13
SC      10-745   10-778   10-781   10-787   10-809#
SC04     4-206#
SC1      4-204#
SC10     4-207#
SC11    10-877   10-885   10-946#
SC12    10-843   10-854   10-965#
SC13    10-833   10-836   10-975#
SC2      4-205#
SC20     4-208#
SC3     10-824#  10-828
SC4     10-826#  10-857   10-869   10-871   10-920   10-936   10-945   10-964   10-:02
SC5     10-825   10-831#
SC6     10-853   10-872#
SC6A    10-851   10-897#
SC7     10-887   10-900   10-902   10-912   10-922#
SC8     10-866   10-880   10-890   10-896   10-905   10-935   10-937#  10-974
SCMND    9-I96    9-K48#
SCOPE    4-81#
SDETAL   9-G88    9-H09    9-H32#
SEARCH   4-252#
SECLMT   6-0#     8-244    9-L71    9-N72
SECOND   6-0#     8-82*    9-I20    9-I31*   9-I32    9-I34*   9-T85*   9-T91*
SEEK     4-244#
SEEKFG  10-132#  10-226   10-454
SELDRV   4-255#
SELPAR   9-114    9-?12#   9-?95
SET.IE  10-255   10-286   10-397   10-471   10-685   10-818   10-<39#
SETFMT   4-254#
SETVEC   8-118    8-120    8-122    8-130    8-137    8-143#
SHDTYP   9-G81    9-H06    9-H19#
SIXTEE   6-0#     8-79*    9-I28*   9-I30*
SIZMEM   8-90#    9-T99
SKI      4-233#
SKIER    9-365    9-766#
SLASH    9-N28    9-N52    9-N67    9-002   16-16#
SPOTCK   9-389    9-473    9-517    9-566    9-607    9-648    9-670    9-B30#
SRCHWT  10-86#   10-424*  10-530*  10-950   10-959*
STACK    4-81#    8-13     8-77
START    4-270    8-3#     9-G68
START1   4-268    8-6#
START2   8-4      8-9#
STATHD   9-H23   15-19#
STATIN   6-0#     8-72*    9-173    9-175*   9-I50*
STATIS   9-157    9-=22#
```

```
STKLMT   4-81#
STNDAT   6-0#     9-:77    9-:83    9->73    9->81
STO      10-:19   10-:39#
STO1     10-:51#
STO2     10-:50   10-:52   10-:54   10-:68#
STO3     10-:70   10-:76#
STO5     10-:82#
STO6     10-:72   10-:84#
STO7     10-:74   10-:92#
STO8     10-:91   10-:98#
STO9     10-:48   10-:58   10-:67   10-:75   10-:79   10-:81   10-:83   10-:89   10-:96   10-;00#
SUMARY   9-39     9-H04#   9-K57    9-P41
SVRH70   10-607   10-768   10-784   10-842   10-892   10-910   10-957   10-991   10-:61   10-:99   10-<07#
SWO      4-81#    9-184    9-?13    9-J00
SWOO     4-81     4-81#
SWO1     4-81     4-81#    9-219    9-887
SWO2     4-81     4-81#    8-148    9-171
SWO3     4-81     4-81#
SWO4     4-81     4-81#    9-A91    9-061    9-P41
SWO5     4-81     4-81#    9-C38    9-C99
SWO6     4-81     4-81#    9-A10
SWO7     4-81     4-81#    9-:35
SWO8     4-81     4-81#
SWO9     4-81     4-81#
SW1      4-81#
SW10     4-81#    9-B71
SW11     4-81#
SW12     4-81#
SW13     4-81#    9-B74    9-T68
SW14     4-81#
SW15     4-81#    9-275    9-F33    9-F58
SW2      4-81#
SW3      4-81#    9-936    9-;73
SW4      4-81#
SW5      4-81#    9-D32
SW6      4-81#
SW7      4-81#    9-186
SW8      4-81#
SW9      4-81#
SWR      5-0#     8-13     8-13     8-13*    8-13*    8-13*    8-14     8-51     8-148    9-171    9-184    9-186    9-219    9-275
         9-887    9-936    9-:35    9-;73    9-?13    9-A10    9-A91    9-B71    9-B74    9-C38    9-C99    9-D32    9-F33    9-F58
         9-J00    9-061    9-P41    9-Q96    9-S74    9-T68    9-T68    9-T68    9-T68    9-T76    9-T76*   9-U05    9-U05*
SWREG    4-268#   8-13     8-14     8-51     9-S74    9-U05    9-U05
SWTIM    9-239    9-282#
SYSTAT   8-151    15-14#
T        9-D51    9-D61    14-5#
TAB.XY   4-276#   5-0
TABLE    9-L73    16-39#
TABLEO   16-39    16-54#
TABLE1   16-42    16-54#
TABLE2   15-42    16-54#
TABLE3   16-42    16-54#
TABLE4   16-42    16-54#
TABLE5   16-42    16-54#
TABLE6   16-42    16-54#
TABLE7   16-42    16-54#
```

```
TBITVE   4-81#
TD      10-743    10-752#
THEAD    9-?20     9-?25#
TIMER   10-137#   10-343*   10-466*   10-610*   10-663*   10-755*   10-873*   10-918*   10-961*   10-976*   10-998*   10-:15    10-:17*   10-:80*
        10-:86*   10-:92*
TKVEC    4-81#     9-S55*    9-S56*
TPVEC    4-81#
TRAPVE   4-81#     8-13*     8-13*
TRE      4-93#
TRFER    9-324     9-746#
TRKLMT   6-0#      8-243     9-977     9-?60     9-?65     9-?67     9-G33     9-L69     9-L74     9-L75     9-L95*    9-L98*    9-M47     9-N57
TRNSWT  10-78#    10-488*   10-635    10-638*   10-651    10-669*   10-678*   10-757    10-758*   10-:55    10-:65*
TRTVEC   4-81#
TST1     8-7#
TYPDS    9-N27     9-N51     9-N66     9-001     9-P41     9-U15#
TYPE     8-14      8-28      8-34      8-39      8-45      8-46      8-123     8-128     8-129     8-131     8-132     8-151     8-152     8-154
         8-162     8-165     8-168     8-171     8-178     8-180     8-181     8-190     8-196     8-206     8-257     9-37      9-38      9-58
         9-59      9-61      9-178     9-248     9-260     9-261     9-262     9-267     9-268     9-269     9-271     9-272     9-293     9-B73
         9-G63     9-G89     9-H36     9-H41     9-H49     9-H49     9-H57     9-H65     9-H73     9-H73     9-H73     9-H73     9-I09     9-I14
         9-I19     9-I64     9-I70     9-J19     9-J21     9-J85     9-J87     9-K69     9-K70     9-K73     9-K74     9-K75     9-K80     9-K83
         9-K85     9-K87     9-L62     9-M09     9-M84     9-M86     9-M87     9-M89     9-N10     9-N20     9-N26     9-N28     9-N42     9-N50
         9-N52     9-N58     9-N65     9-N67     9-N73     9-N96     9-002     9-010     9-027     9-028     9-029     9-031     9-045     9-046
         9-047     9-048     9-P41     9-P41     9-P41     9-P41     9-P41     9-P41     9-P41     9-P41     9-Q55     9-S69     9-S70     9-S78
         9-S79     9-S86     9-T09     9-T15     9-T20     9-T24     9-T29     9-T34     9-T35     9-T37     9-T40     9-T44     9-T68     9-T68
         9-T68     9-T70     9-T70     9-T70     9-T70     9-T70     9-T70     9-T70     9-T72     9-T76     9-U01     9-U03     9-U05     9-U05
         9-U05     9-U05     9-U05     9-U05     9-U15#   16-63     16-71     16-101    16-103    16-106    16-112    16-115
TYPOC    9-T70     9-T70     9-U05     9-U15#   16-105    16-114
TYPON    9-U15#
TYPOS    8-31      8-42      8-153     9-60      9-273     9-H35     9-J86     9-K79     9-M88     9-030     9-050     9-P41     9-P41     9-U15#
TYPRI4   9-H20     9-H22     9-Q83#
UCPAR    9-235     9-260#
ULDFLG  10-105#   10-281*   10-369    10-384*   10-567*   10-633*   10-670*   10-839    10-925    10-927*   10-946    10-948*   10-:64*
UNIT     6-0#      9-208*    9-273     9-C33     9-P41     9-P41     9-P41*
UNS      4-166#
UNSAF    9-318     9-797#
UNTASN   9-J34     9-J49    15-8#
UNTMSG   9-59      9-272     9-M87     9-029     9-048     9-P41     9-P41    15-4#
UNTNOT   9-K41     9-K60    15-7#
UNTOFF   8-168     9-K12    15-5#
UNTON    8-180    15-6#
UPE      4-117#
US1      4-104#
US2      4-105#
US4      4-106#
VV       4-139#
WAIT     6-0#      9-29      9-80      9-82      9-94      9-132
WATPAK   9-J17     9-L02#
WC.HK   10-770    10-791#
WCE      4-118#
WCF      4-157#
WCFER    9-342     9-778#
WCHKX    9-U20#   10-769    10-790
WCKD     4-256#    6-0
WCKER    9-321     9-457#
WCKHD    4-257#    6-0
WCSEL    6-0#      9-@00    16-7
```

```
WLEER    9-348     9-706#
WRL      4-144#
WRT.AD  10-;64*   10-;74*   10-;77#   13-4
WRT.R1  10-;75#   10-;95
WRT.R2  10-;61*   10-;94#
WRT.R3  10-;70    10-;81    10-;84    10-;90    10-;96#
WRT.R4  10-;86    10-;98#
WRT.R5  10-;97    10-;99#
WRT.RM  10-315    10-319    10-497    10-501    10-505    10-513    10-523    10-527    10-538    10-545    10-555    10-569    10-579    10-600
        10-615    10-628    10-864    10-894    10-933    10-;61#
WRT.WD  10-;62*   10-;66    10-;73*   10-;76#   13-4
WRTDAT   4-258#    6-0       9-B12
WRTHD    4-259#    6-0
WRTPK    8-248     9-112     9-A63#
WRTPK1   9-A65     9-A67     9-A71#
WRTPK2   9-A70     9-B06#
WRTPK3   9-A69     9-B05     9-B10#
WRTPK4   9-B09     9-B15#
WRTPK5   9-A88     9-B17#
XWCNT    9-?30     9-?34     9-?87#
XXDP     6-0#      8-19*     8-22*     8-23      8-25*     8-30      8-41      8-174     8-176     8-224     8-226     9-J35     9-J37     9-J50
         9-J52
Y       15-32#
ZEROS    6-0       6-0#
ZROIND   6-0#      9-900*    9-946*    9-952*    9-956
```

```
$$CMRE   4-822#
$$CMTM   4-822#   5-0
$$ESCA   4-81#
$$NEWT   4-81#    8-7
$$SET    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15#   9-U16
$$SETM   8-13     8-13#
$$SKIP   4-81#
.$ACT1   4-55#    4-272
.$APTB   4-59#    5-0      5-0#
.$APTH   4-59#    4-275
.$APTY   4-59#    9-T74
.$CATC   4-56#    4-268
.$CMTA   4-56#    4-822
.$DB2D   4-57#    9-U11
.$DB2O   4-57#    9-U13
.$EOP    4-59#    9-P41
.$ERRO   4-56#    9-T68
.$ERRT   4-56#    9-T70
.$POWE   4-59#    9-T76
.$RAND   4-58#    9-U07
.$READ   4-55#    9-U05
.$SAVE   4-57#    9-U09
.$SIZE   4-57#   16-85
.$TRAP   4-57#    9-U15
.$TYPD   4-56#    9-U03
.$TYPE   4-55#    9-T72
.$TYPO   4-56#    9-U01
.EQUAT   4-55#    4-81
.HEADE   4-55#    4-63
.SETUP   4-55#    4-266
.SWRHI   4-55#    4-64
.SWRLO   4-55#    4-64#    4-65     4-66     4-67     4-69     4-71     4-76     4-78     4-79
A        9-H84#   9-H99    9-I00    9-I01    9-I02    9-I03
CKCHR    4-16#    9-R89    9-R97
CKDIG    4-26#    9-N41    9-N57    9-N72    9-O07
CKNUM    4-39#
COMMEN   4-81#
ENDCOM   4-81#
ERENTR   9-T60#   9-T68
ERRCAL   9-U23#  10-817   10-868   10-870   10-;32   10-;91
ERROR    4-8#
ESCAPE   4-81#
GETPRI   4-81#   16-85
GETSWR   4-58#    4-81#    8-14     8-14#    8-51
MORETA   4-278#   5-0
MULT     4-81#
NEWTST   4-81#    8-7
POP      4-81#    9-:14    9-@59    9-B59    9-C58    9-H81    9-L48    9-L99    9-M44    9-M95    9-N84    9-T74    9-T74    9-T76
         9-T76    9-U03    9-U07    9-U09
PUSH     4-81#    9-:07    9-@40    9-B30    9-B94    9-L19    9-L92    9-M35    9-M39    9-N05    9-T74    9-T74    9-T74    9-T76
         9-T76    9-U03    9-U07    9-U09
REPORT   4-81#
SETPRI   4-81#
SETTRA   9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15    9-U15#   9-U16
SETUP    4-81#    8-13
SKIP     4-81#
```

```
SLASH    4-81#
STARS    4-81#    4-272    4-275    4-275    4-275    5-0      5-0      5-0      8-7      9-B62    9-B66    9-F98    9-G02    9-P41
         9-T68    9-T70    9-T72    9-T74    9-T76    9-T76    9-U01    9-U03    9-U05    9-U05    9-U05    9-U07    9-U09    9-U11
         9-U13    9-U15    10-14    16-85
SWRSU    4-81#    8-13     8-13#
TRMTRP   9-U15#   9-U17
TYPBIN   4-81#
TYPDEC   4-81#    9-N27    9-N51    9-N66    9-P41
TYPNAM   4-81#    8-14
TYPNUM   4-81#
TYPOCS   4-81#    8-153    9-60     9-273    9-H35    9-J86    9-K79    9-M88    9-030    9-050    9-P41    9-P41
TYPOCT   4-81#    9-T70    9-T70    9-U05
TYPTXT   4-81#    8-28     8-39     8-45     8-46
XXEP     9-078#   9-P41
```