RM03

RM03 PERFORMANCE EX.

MD-11-DZRMB-A

EP-DZRMB-A-DL-A

COPYRIGHT © 1977

FICHE 1 OF 2

OCT 1977

digital

MADE IN USA

RM03

RM03 PERFORMANCE EX.

MD-11-DZRMB-A

EP-DZRMB-A-DL-A

COPYRIGHT © 1977

FICHE 2 OF 2

OCT 1977

digital

MADE IN USA

B01

EOF1DZRMABSEQ	00010000	770804	PDP10 411	HDR1DZRMBASEQ	00010000	770804

```
                                .REM    0
```

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

PRODUCT CODE: MAINDEC-11-DZRMB-A-D

PRODUCT NAME: RM03 PERFORMANCE EXERCISER

DATE CREATED: AUGUST 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. CHEN

COPYRIGHT (C) 1977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THE INFORMATION IN THIS STATEMENT IS SUBJECT TO CHANGE
WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A
COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

ACTUAL DISTRIBUTION OF THE SOFTWARE DESCRIBED IN THIS
DOCUMENT WILL BE SUBJECT TO TERMS AND CONDITIONS TO
BE ANNOUNCED ON SOME FUTURE DATE BY DIGITAL EQUIPMENT
CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR
RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS
NOT SUPPLIED BY DEC.

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

CONTENTS

1. ABSTRACT

2. REQUIREMENTS

    2.1 EQUIPMENT
    2.2 MEDIA
    2.3 PRELIMINARY PROGRAMS

3. OPERATING THE PROGRAM

    3.1 LOADING THE PROGRAM
    3.2 STARTING THE PROGRAM
    3.3 RESTARTING THE PROGRAM
    3.4 PROGRAM CONTROL
    3.5 PASS/TEST TERMINATION
        3.5.1 PASS TERMINATION
        3.5.2 TEST TERMINATION
    3.6 RUN TIME
        3.6.1 DATA TRANSFER MODE
        3.6.2 SEEK VERIFICATION MODE
    3.7 UNIBUS & VECTOR ADDRESSES
    3.8 DUAL PORT OPERATION
    3.9 APT
    3.10 APT ENVIRONMENTAL TABLE DEFINITIONS

4. CONTROLLING THE PROGRAM

    4.1 DATE & OPERATOR IDENTIFICATION
    4.2 PARAMETERS
        4.2.1 PROGRAM CONTROL PARAMETERS
        4.2.2 PERIPHERIAL DEVICE ADDRESSES
        4.2.3 PARAMETERS FOR THE FIRST OPERATION
    4.3 SWITCH REGISTER SETTINGS
    4.4 KEYBOARD COMMANDS
        4.4.1 'T' COMMAND
        4.4.2 'D' COMMAND
        4.4.3 'S' COMMAND
        4.4.4 'W' COMMAND
        4.4.5 'R' COMMAND
        4.4.6 'WT' COMMAND
        4.4.7 GENERAL COMMAND INFORMATION

5. PERFORMANCE SUMMARY TYPEOUT

    5.1 PERFORMANCE SUMMARY TYPEOUT EXPLANATION
    5.2 HARD/SOFT ERROR DEFINITIONS
        5.2.1 HARD ERRORS
        5.2.2 SOFT ERRORS

6. DATA CHECKING & ERROR RECOVERY

    6.1 DATA BUFFER COMPARISON
    6.2 VERIFICATION OF DATA WRITTEN

6.3 SECTOR REFORMATTING
6.4 BAD TRACK/SECTOR FLAGGING

7. ERROR MESSAGES

7.1 ERROR DESCRIPTION LINES
7.2 DETAIL ERROR LINES

8. PROGRAM DESCRIPTION

8.1 HOW THE PROGRAM OPERATES
8.2 DUAL PORT OPERATION
8.3 HOW VARIABLES ARE SELECTED FOR EACH OPERATION
8.4 DATA PATTERNS

9. RM03 SOFTWARE DRIVER DOCUMENT

1. ABSTRACT

THE RM03 PERFORMANCE EXERCISER PROGRAM IS DESIGNED TO PERFORM AN INTERACTIVE TEST ON RM03 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 RM03 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.

2. REQUIREMENTS

2.1 EQUIPMENT

REQUIRED

PDP-11 PROCESSOR
16K MEMORY (20K IF THE PROGRAM IS INCLUDED IN AN 'XXDP' CHAIN
TELETYPE
PROGRAM LOADING DEVICE
KW11-L OR KW11-P CLOCK
RH11 OR RH70 WITH 1 RM03 DISK DRIVE

OPTIONAL

ADDITIONAL MEMORY TO A MAXIMUM OF 28K
1 TO 7 ADDITIONAL RM03'S ON THE SAME RH11 OR RH70

2.2 MEDIA

THE RM03 PERFORMANCE EXERCISER PROGRAM REQUIRES FORMATTED DISK PACKS GENERATED BY THE RM03 FORMATTER PROGRAM (MD-11-DZRMA-A). 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

RM03 DISKLESS DIAGNOSTIC

RM03 FUNCTIONAL TEST

3. OPERATING THE PROGRAM

3.1 THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA USING THE ASSOCIATED LOADER. THE PROGRAM MAY BE INCLUDED IN AN 'XXDP' CHAIN. IF THE PROGRAM IS BEING RUN ON A PROCESSOR WITH 16K, THE 'XXDP' LOADER WILL NOT BE PRESERVED. THE PROGRAM MUST BE RUN ON A SYSTEM WITH 20K OR MORE TO PRESERVE THE 'XXDP' LOADER. THE 'ABSOLUTE' LOADER WILL BE PRESERVED IN A 16K SYSTEM, HOWEVER.

3.2 THE PROGRAM STARTS AT LOCATION 200(8). PARAMETERS NOT INCLUDED IN THE TELETYPE DIAGLOGUE GROUP MUST BE CHANGED BEFORE THE PROGRAM IS STARTED.
THE PROGRAM PERFORMS "TEST" OPERATION TO ALL DRIVES UNDER TESTING AT THE 200(8) STARTING LOCATION AUTOMATICALLY.

3.3 START THE PROGRAM AT LOCATION 204(8) IF THE RH11 OR THE RH70 IS NOT AT ADDRESS 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 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 RM03'S 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.

3.5.1 PASS TERMINATION

END OF PASS FOR A SINGLE DRIVE MAY BE DETERMINED BY EITHER OF THE FOLLOWING CONDITIONS. THE END OF PASS CONDITION USED IS DETERMINED BY PARAMETER 'ENDING'.

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.

3.5.2 TEST TERMINATION

THE TEST FOR A DRIVE IS TERMINATED (SW<04> = 0) 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 DECIMAL) 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 - APPROXIMATELY 3.6 HRS (TO REACH 1.875 X 10↑8 WORDS)
TO
8 DRIVES - APPROXIMATELY 16 HRS (FOR ALL DRIVES TO REACH 1.875 X 10↑8 WORDS)

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.)

UNIT	UNIBUS ADDRESS	VECTOR 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 RM03 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; 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' ('RO'). TYPING A 'RO' 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:

ENTER PARAMETERS:

THE OPERATOR MUST ENTER A 'Y' TERMINATED BY A CARRIAGE RETURN <CR> 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	5	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 "TEST" COMMAND
                                                                               IF PARAMETER=1,SEQUENTIAL
                                                                               DATA BLOCK IS USED IN
                                                                               "TEST" 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 (10) 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(8) 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).
```

LOC	TAG	INITIAL VALUE	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 'F~2COD')
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 NORMAL END OF TEST 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 SUMMARY TYPEOUTS.
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 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS IN KEYBOARD ENTRY MODE, OR IS AT A HIGHER PRIORITY PROCESSING AN RM03 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

```
        'SWR = NNNNNN    NEW ='
```

EACH TIME SWITCH 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).

AFTER THE PROGRAM HAS BEEN INITIALIZED, THE FOLLOWING MESSAGE
WILL BE TYPED:

```
'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 ADDRESS LIMITS, DRIVE I.D,
AND BAD LOCATION ADDRESS ENTRIES FOR THE DRIVE BEING REFERENCED.

THE PROGRAM WILL FIRST ASK FOR ADDRESS LIMITS WITH THE FOLLOWING
TYPEOUT:

```
'ENTER ADDRESS LIMITS FOR DRV #N / RM03
```

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

```
                DEFAULT   VALUE
```

NAME	BASE	VALUE	RANGE	FUNCTION
MINCYL	10	0	0 - 822	THE MINIMUM CYLINDER ADDRESS
MAXCYL	10	822	0 - 822	THE MAXIMUM CYLINDER ADDRESS
MINTRK	10	0	0 - 4	THE MINIMUM TRACK ADDRESS
MAXTRK	10	4	0 - 4	THE MAXIMUM TRACK ADDRESS
MINSEC	10	0	0 - 31	THE MINIMUM SECTOR ADDRESS
MAXSEC	10	31	0 - 31	THE MAXIMUM SECTOR ADDRESS

NOTE: 1. THE ADDRESS LIMITS ARE IN DECIMAL

2. 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 'MIN' ADDRESS AND THE UPPER PHYSICAL LIMIT FOR THAT ADDRESS AND BETWEEN 'O' AND THE VALUE IN 'MAX'.

EACH COMMAND, EXCEPT THE 'S' AND THE 'D' COMMANDS, WILL ASK FOR A DRIVE IDENTIFICATION ENTRY. THE DRIVE IDENTIFICATION ENTRY REQUEST IS MADE BY THE FOLLOWING MESSAGE:

'ENTER I.D. FOR DRV #N:'

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.

4.4.1 'T' COMMAND

USED TO ASSIGN A DRIVE(S) FOR 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 'SA' IS USED ONLY DRIVES BEING TESTED WILL
           BE DISPLAYED.

        2. 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'.

        3. 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 RM03 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:  WA<CR> - WRITE DATA PACKS ON ALL AVAILABLE DRIVES.
                                                  W0<CR> - GENERATE A DATA PACK ON DRIVE 0.

                                        NOTES:  1.  DATA PATTERNS GENERATED BY THE RM03 FORMATTER PROGRAM
                                                    (MD-11-DZRMA-A) 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:  RA<CR> - READ THE PACKS ON ALL OF THE ONLINE DRIVES.
                                                  R0<CR> - READ THE PACK ON DRIVE 0.

                                        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"TEST" COMMAND

                                        FORMAT: WTN<CR>
```

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

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

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)
?UNIT N OFFLINE	T, W, R,WT
?UNIT N NOT ASSIGNED	D, S
?UNIT N ALREADY ASSIGNED	T, W, R,WT
?UNIT N NOT PRESENT	T, W, R,WT
?UNIT N UNSAFE	T, W, R,WT
?UNIT N NOT AN RM03	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

6.4 BAD TRACK/SECTOR FLAGGING

SINCE THE RM03 SUBSYSTEM HAS AN AUTOMATIC BAD TRACK HANDLING CAPABILTIY, THE MULTIDRIVE EXERCISER ALLOWS TO IDENTIFY UP TO 16 BAD TRACK/SECTOR LOCATIONS WHEN THE DRIVE IS ASSIGNED FOR TEST. (SEE SECTION 4.4 FOR ADDITIONAL INFORMATION.)

IF ONE OF THE FOLLOWING ERRORS OCCURS AT A LOCATION IDENTIFIED BY THE OPERATOR, 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' & 'HCRC', 'HCE' & 'HCRC', 'HCRC')

OPTIONALLY, THE OPERATOR MAY REQUEST AN ERROR REPORT FOR FLAGGED AREAS. (PARAMETER 'NOTPRT' MUST BE SET TO 0 AT STARTUP.) THE PROGRAM WILL IDENTIF/ ERROR MESSAGES ASSOCIATED WITH THE PACK; THESE ERRORS WILL NOT BE ADDED TO THE ERROR TOTALS MAINTAINED BY THE PROGRAM.

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 RM03 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 RM03 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 REIGSTER IS EITHER'0' OR > 40066 (8)
                        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

TT:TT:TT (DESCRIPTION OF ERROR)

'TT:TT:TT' IS THE TIME SINCE THE PROGRAM WAS STARTED. TT:TT:TT IS GIVEN IN 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 RH11/RM03 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 RM03 REGISTERS. THE CONTENTS OF THIS WORD IDENTIFY HOW THE ERROR WAS PROCESSED BY THE RM03 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 RM03 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 RH11/RM03 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 RM03, AND IS ONLINE. THE ASSIGNMENT ROUTINE THEN ISSUES A 'READIN PRESET' INSTRUCTION, SETS 'FMT22', 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 (10); 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

```
1805    000020  177757  172666  173777  153333  177777  000000
1806    000040  177737  155555  175777  066667  177777  000000
1807    000100  177677  172666  176777  153333  177777  000000
1808    000200  177577  155555  177377  066667  177777  000000
1809    000400  177377  172666  177577  153333  177777  000000
1810    001000  176777  155555  177677  066667  177777  000000
1811    002000  175777  172666  177737  153333  177777  000000
1812    004000  173777  155555  177757  066667  177777  000000
1813    010000  167777  172666  177767  153333  177777  000000
1814    020000  157777  155555  177773  066667  177777  000000
1815    040000  137777  172666  177775  153333  177777  000000
1816    100000  077777  155555  177776  066667  177777  000000
```

9.1 RH70(11)/RM03 DRIVER

THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH70/RM03 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 RM03
=0	OFFLINE RM03, DRIVE IS NOT AN RM03, OR NONEXISTENT DRIVE
<0	UNSAFE RM03

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

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

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

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

CALL:

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

*DPB (DATA PARAMETER BLOCK)

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

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

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

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

```
                        .WORD   0               ;STATUS/ERROR INDICATOR

                        ALTERNATE DPB SETUP

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

9.5 RH70/RM03 REGISTERS

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

9.6 COMMANDS PERFORMED BY THE DRIVER

COMMAND	CODE	COMMAND TYPE
NO OPERATION	101	N
UNLOAD	103	N
SEEK	105	P
RECALIBRATE	107	P
DRIVE CLEAR	111	N
RELEASE	113	N
OFFSET	115	P

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

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

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.  (RH70/RM03
                                                        REGISTERS WERE NOT SAVED)

                                        (2) =>          REQUEST QUEUE HAS BEEN EMPTIED.  THE DRIVER
                                                        ISSUED A "DRIVE CLEAR" TO THE DRIVE.
                                                        NOTE: ALL RH70/RM03 REGISTERS ARE SAVED
                                                        AS PER DPB+14 BEFORE THE "DRIVE CLEAR".

                                        (3) =>          REQUEST QUEUE HAS BEEN EMPTIED.  THE
                                                        DRIVER ISSUED A MASSBUS INIT.  ALL
                                                        RH70/RM03 REGISTERS FOR THE DRIVE WERE
                                                        SAVED AS PER DPB+14 BEFORE THE INIT.

                                        (4) =>          A "RECALIBRATE" SHOULD BE ISSUED
                                                        BEFORE ANY OTHER COMMAND.

                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)

                                        ;
```

```
                                .TITLE  MD-11-DZRMB A0/00, RM03 PERFORMANCE EXERCISER
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY C. CHEN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
                                .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 WHEN NORMAL END OF TEST REACHED
                                ;*         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 & SW7 SET, DISPLAY
                                ;*                          REMAINDER OF BUFFER
                                ;*         2            A. DON'T TYPE SUBSYSTEM STATUS WHEN PROGRAM STARTED
                                ;*                      B. DON'T TYPE PERFORMANCE SUMMARY
                                ;*         1            INHIBIT DATA COMPARSION AFTER READ ORDERS
                                ;*         0            READ ONLY MODE
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

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

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
```

```
000003          R3=     %3              ;;GENERAL REGISTER
000004          R4=     %4              ;;GENERAL REGISTER
000005          R5=     %5              ;;GENERAL REGISTER
000006          R6=     %6              ;;GENERAL REGISTER
000007          R7=     %7              ;;GENERAL REGISTER
000006          SP=     %6              ;;STACK POINTER
000007          PC=     %7              ;;PROGRAM COUNTER

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

                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000          SW15=   100000
040000          SW14=   40000
020000          SW13=   20000
010000          SW12=   10000
004000          SW11=   4000
002000          SW10=   2000
001000          SW09=   1000
000400          SW08=   400
000200          SW07=   200
000100          SW06=   100
000040          SW05=   40
000020          SW04=   20
000010          SW03=   10
000004          SW02=   4
000002          SW01=   2
000001          SW00=   1
                .EQUIV  SW09,SW9
                .EQUIV  SW08,SW8
                .EQUIV  SW07,SW7
                .EQUIV  SW06,SW6
                .EQUIV  SW05,SW5
                .EQUIV  SW04,SW4
                .EQUIV  SW03,SW3
                .EQUIV  SW02,SW2
                .EQUIV  SW01,SW1
                .EQUIV  SW00,SW0

                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000          BIT15=  100000
040000          BIT14=  40000
020000          BIT13=  20000
010000          BIT12=  10000
004000          BIT11=  4000
002000          BIT10=  2000
001000          BIT09=  1000
000400          BIT08=  400
000200          BIT07=  200
```

```
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
176700                          ABASE=176700
000254                          AVECT1=254

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

                        .SBTTL  RM03 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  RM03 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                       ;RM03 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)

                ;RM03 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  RM03 DRIVER COMMANDS

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

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

```
                000163          WRTHD   =       163             ;WRITE HEADER & DATA
                000171          RDDAT   =       171             ;READ DATA
                000173          RDHD    =       173             ;READ HEADER & DATA

                                .SBTTL  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  003632                  JMP     @#START1        ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  003622                  JMP     @#START         ;CHANGE THE RH11 UNIBUS ADDRESS
                                                                ;AFTER INITIAL START

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
                000210                  $SVPC=.                 ;SAVE PC
                000046                  .=46
000046  027260                          $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  001204                  $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 ADDITIONAL UNIT
001112  000032                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                001114                  TAB.XY=.
```

```
                                        .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  177607  000377                  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001200     077                          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001201     015                          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001202  000012                          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                        ;;*****************************************************************
                                        .SBTTL  APT MAILBOX-ETABLE

                                        ;;*****************************************************************
                                        .EVEN
001204                                  $MAIL:                          ;;APT MAILBOX
001204  000000                          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001206  000000                          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001210  000000                          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001212  000000                          $PASS:  .WORD   APASS           ;;PASS COUNT
001214  000000                          $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001216  000000                          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001220  000000                          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001222  000000                          $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001224                                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
```

```
001224     000                  $ENV:    .BYTE   AENV     ;;ENVIRONMENT BYTE
001225     000                  $ENVM:   .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
001226  000000                  $SWREG:  .WORD   ASWREG   ;;APT SWITCH REGISTER
001230  000000                  $USWR:   .WORD   AUSWR    ;;USER SWITCHES
001232  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
001234     000                  $MAMS1:  .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
001235     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
001236  000000                  $MADR1:  .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
                                ;*                          MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001240     000                  $MAMS2:  .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
001241     000                  $MTYP2:  .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
001242  000000                  $MADR2:  .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
001244     000                  $MAMS3:  .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
001245     000                  $MTYP3:  .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
001246  000000                  $MADR3:  .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
001250     000                  $MAMS4:  .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
001251     000                  $MTYP4:  .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
001252  000000                  $MADR4:  .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
001254  000254                  $VECT1:  .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001256  000000                  $VECT2:  .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001260  176700                  $BASE:   .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001262  000000                  $DEVM:   .WORD   ADEVM    ;;DEVICE MAP
001264  000000                  $CDW1:   .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
001266  000000                  $CDW2:   .WORD   ACDW2    ;;CONTROLLER DESCRIPTION WORD#2
001270                          $ETEND:
                                .MEXIT
        000015                  CR       =       15
        000012                  LF       =       12
001270  176700                  $RMADR:  .WORD   176700          ;FIRST ADDRESS OF RH11/RM03 REGISTERS
001272  000254                  $RMVEC:  .WORD   254             ;RM03 VECTOR ADDRESS
001274  172540                  $LKCSR:  .WORD   172540          ;ADDR OF KW11-P STATUS REGISTER
001276  172542                  $LKCSB:  .WORD   172542          ;ADDR OF KW11-P COUNTER BUFFER
001300  000104                  $LPVEC:  .WORD   104             ;ADDR OF KW11-P VECTOR
001302  177546                  $LKS:    .WORD   177546          ;ADDR OF KW11-L STATUS REGISTER
001304  000100                  $LLVEC:  .WORD   100             ;ADDR OF KW11-L VECTOR
001306  177777                  PCLOCK:  .WORD   -1              ;'0' IF KW11-P IS ON SYSTEM
001310  177777                  CLKFLG:  .WORD   -1              ;'0' IF A CLOCK IS AVAILABLE
001312  000074                  HZ:      .WORD   74              ;74 (8) IF 60 HZ SYSTEM; 62 (8) IF 50 HZ SYSTEM
001314  000000                  STMTIN:  .WORD   0               ;'TYPE STATISTICS' INDICATOR
001316  000000                  PACK:    .WORD   0               ;'W ' COMMAND INDICATOR
001320  000000  000000  000000  DATE:    .WORD   0,0,0,0,0       ;OPERATOR ENTERED DATE
001326  000000  000000
001332  000000  000000  000000  OPERID:  .WORD   0,0,0,0          ;OPERATOR ID
001340  000000
        001216                  DRIVE    =       $UNIT           ;DRIVE # STORAGE: ERRORS 1-5 & 10
001342  000000                  ATTN:    .WORD   0               ;ATTN REG STORAGE: ERRORS 1-5 & 10
```

```
001344  000000                  UNIT:   .WORD   0               ;DRIVE # STORAGE FOR PRINTOUT
000000  000000                  MASK:   .WORD   0               ;STORE RETRY REGISTER MASK
001350  000000  000             RETRY:  .BYTE   0,0             ;STORE RETRY LIMIT IN THE LOWER BYTE
                                                                ;RETRY COUNT IN THE UPPER BYTE
001352  000003                  FAIRNS: .WORD   3               ;MAXIMUM TIME IN 60 HZ VALUE
000000  000000                  LSTAD:  .WORD   0               ;STORE LAST MEMORY ADDRESS HERE
000006  000000                  CHGADR: .WORD   0               ;CHANGE RH11 UNIBUS ADDRESS FLAG
000000  000000                  CFLAG:  .WORD   0               ;'CONTROL C' FLAG
000000  000000                  BADSEC: .WORD   0               ;BAD SECTOR/TRACK FLAG
001354  000000                  HOUR:   .WORD   0               ;HOUR COUNT STORED HERE (MAXIMUM - 999.)
001366  000000                  MINUTE: .WORD   0               ;MINUTE'S COUNT STORED HERE
001370  000000                  SECOND: .WORD   0               ;SECOND'S COUNT STORED HERE
001372  000000                  SIXTEE: .WORD   0               ;TIMER ROUTINE COUNTER (FOR ONE SECOND)
001374  177777                  ZROIND: .WORD   -1              ;ZERO INDICATOR FOR THE DATA COMPARE ROUTINE
001376  000                     FRSTER: .BYTE   0               ;DATA COMPARE ERROR FLAG
                                                                ;IF > 0, PROCESSING 'DOKER' OR CAN'T MATCH PATTERN
                                                                ;IF < 0, MISCOMPARSION FOUND
001377  000                             .BYTE   0               ;MISCOMPARSION OR CAN'T MATCH PATTERN FLAG
                                                                ;IF < 0, ERROR IN BUFFER
001400  000000                  SAVER1: .WORD   0               ;SAVE R1 HERE
001402  000000                  SAVERS: .WORD   0               ;SAVE RS HERE
001404  000000                  ERCTR:  .WORD   0               ;NUMBER OF ERRORS
001406  000000                  LIMIT:  .WORD   0               ;DISPLAY LIMIT
001410  000000                  CMCNT:  .WORD   0               ;WORD COUNT
001412  000000                  CMCYL:  .WORD   0               ;CYLINDER ADDRESS
001414  000                     CMSEC:  .BYTE   0               ;SECTOR ADDRESS
001415  000                     CMTRK:  .BYTE   0               ;TRACK ADDRESS
001416  000000                  ECBIT:  .WORD   0               ;ERROR BURST BIT OFFSET
001420  000000                  ECSEC:  .WORD   0               ;ERROR BURST WORD OFFSET (RELATIVE TO SECTOR)
001422  000000                  ECMSK0: .WORD   0               ;CORRECTION MASK FOR FIRST ERROR WORD
001424  000000                  ECMSK1: .WORD   0               ;CORRECTION MASK FOR SECOND ERROR WORD
001426  000000                  ECWRD:  .WORD   0               ;LOCATION OF FIRST ERROR WORD
001430  000000                  ECGD:   .WORD   0               ;GOOD DATA, FIRST WORD
001432  000000                  ECBAD0: .WORD   0               ;BAD DATA, FIRST WORD
001434  000000                  ECWRD1: .WORD   0               ;LOCATION OF SECOND ERROR WORD
001436  000000                  ECGD1:  .WORD   0               ;GOOD DATA, SECOND WORD
001440  000000                  ECBAD1: .WORD   0               ;BAD DATA, SECOND WORD
001442  000037                  SECLMT: .WORD   31.             ;SECTOR ADDRESS LIMIT
001444  000004                  TRKLMT: .WORD   4.              ;TRACK ADDRESS LIMIT
001446  001465                  CYLIMT: .WORD   821.            ;CYLINDER ADDRESS LIMIT FOR RM03

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

                                .SBTTL  COMMON PARAMETERS

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

001450  002740                  ENDCON: .WORD   002740          ;1.875X10↑8 WORDS (10) [3X10↑9 BITS]
001452  005455                          .WORD   005455          ;MSW
001454  143300                  ENDSEK: .WORD   143300          ;3 X 10↑6 SEEKS  (LSW)
001456  000055                          .WORD   55              ;MSW
001460  000001                  PASCNT: .WORD   1               ;NUMBER OF PASSES TO END OF TEST
001462  000000                  MAXDL:  .WORD   0               ;MAXIMUM DATA TRANFER SIZE IN WORDS
                                                                ;(FILLED BY PROGRAM AT STARTUP OR BY OPERATOR
                                                                ;DURING PARAMETER ENTRY DIALOG.)
001464  000144                  MAXER:  .WORD   100.            ;MAXIMUM ERRORS - 100(10)
```

```
001466  000005  000000          INTRVL: .WORD   5,0          ;FIRST WORD IS THE PERFORMANCE TYPEOUT INTERVAL
                                                             ;(IN MINUTES).  SECOND WORD IS THE INTERVAL
                                                             ;COUNTER
                                                             ;COUNTER.  UPPER BYTE IS VALUE.
001472  000004                  CMPLMT: .WORD   4            ;NUMBER OF COMPARE ERRORS TYPED OUT
001474  000001                  FORMAT: .WORD   1            ;IF NOT EQ 0, ALLOW WRITE HEADER & DATA ORDERS
                                                             ;IF EQ 0, DO NOT ALLOW WRITE HEADER & DATA ORDERS
001476  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
001500  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
001502  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.
001504  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.
001506  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.
001510  000000                  PATTEN: .WORD   0            ;IF EQ 0,RANDOMLY SELECT DATA PATTERN
                                                             ;IF NOT EQ 0,SELECT ONE SET OF PATTERN
                                                             ;POINTED BY THE "PATTEN".
001512  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

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

001514  000010                  BEGPAT: .WORD   10           ;STARTING PATTERN CODE [RANGE 1 - 17 (OCTAL)]
001516  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')
001520  000400                  BEGSIZ: .WORD   400          ;STARTING RECORD SIZE [RANGE 4 - MAXMEM]
```

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

                                        .SBTTL  TABLES, CONSTANTS, AND VARIABLE LOCATIONS

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

001522  000000  000000  000000  ORDERQ: .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF DRIVES PERFORMING COMMANDS
001530  000000  000000  000000
001536  000000  000000  000000

001544  000000                  ASNLST: .WORD   0               ;A BIT SET IS AN ASSIGNED DRIVE

001546  000000  000000  000000  DUNIT:  .WORD   0,0,0,0,0,0,0,0,0      ;ADDRESSES OF DRIVES TO BE DEASSIGNED
001554  000000  000000  000000
001562  000000  000000  000000

001570  000000  000000  000000  NEWUNT: .WORD   0,0,0,0,0,0,0,0,0      ;ADDRESSES OF NEWLY ASSIGNED DRIVES
001576  000000  000000  000000
001604  000000  000000  000000

001612  000000  000000  000000  AVAIL:  .WORD   0,0,0,0,0,0,0,0,0      ;LIST OF DRIVES WAITING FOR BUFFERS/PARAMETERS
001620  000000  000000  000000
001626  000000  000000  000000

001634  000000  000000  000000  WAIT:   .WORD   0,0,0,0,0,0,0,0,0      ;LIST OF DRIVES WAITING FOR BUFFERS
001642  000000  000000  000000
001650  000000  000000  000000

001656  000000  000000  000000  PARQ:   .WORD   0,0,0,0,0,0,0,0,0      ;LIST OF DRIVES WAITING FOR NEXT PARAMETERS
001664  000000  000000  000000
001672  000000  000000  000000

001700  000000                  BUFTBL: .WORD   0               ;BUFFER ALLOCATION TABLE ENTRY COUNT
001702  000000  000000                  .WORD   0,0
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  042472                  BLKADR: .WORD   DRIVE0          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 0
002104  042776                          .WORD   DRIVE1          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 1
002106  043302                          .WORD   DRIVE2          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 2
002110  043606                          .WORD   DRIVE3          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 3
002112  044112                          .WORD   DRIVE4          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 4
002114  044416                          .WORD   DRIVE5          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 5
002116  044722                          .WORD   DRIVE6          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 6
002120  045226                          .WORD   DRIVE7          ;ADDRESS OF THE PARAMETER BLOCK FOR DRIVE 7

002122     151                  COMTBL: .BYTE   WCKD            ;WRITE CHECK DATA
002123     153                          .BYTE   WCKHD           ;WRITE CHECK HEADER AND DATA
002124     161                          .BYTE   WRTDAT          ;WRITE DATA
002125     163                          .BYTE   WRTHD           ;WRITE HEADER AND DATA
002126     171                          .BYTE   RDDAT           ;READ DATA
002127     173                          .BYTE   RDHD            ;READ HEADER AND DATA

002130     002                  OPTBL:  .BYTE   2               ;UNLOAD
002131     004                          .BYTE   4               ;SEEK
002132     006                          .BYTE   6               ;RECAL
002133     010                          .BYTE   10              ;DRIVE CLEAR
002134     012                          .BYTE   12              ;RELEASE
002135     014                          .BYTE   14              ;OFFSET
002136     016                          .BYTE   16              ;RETURN TO CENTERLINE
002137     020                          .BYTE   20              ;READIN PRESET
002140     022                          .BYTE   22              ;PACK ACKNOWLEDGE
002141     030                          .BYTE   30              ;SEARCH
002142     050                          .BYTE   50              ;WRITE CHECK DATA
002143     052                          .BYTE   52              ;WRITE CHECK HEADER AND DATA
002144     060                          .BYTE   60              ;WRITE DATA
002145     062                          .BYTE   62              ;WRITE HEADER AND DATA
002146     070                          .BYTE   70              ;READ DATA
002147     072                          .BYTE   72              ;READ HEADER AND DATA
002150     377                          .BYTE   -1              ;TERMINATOR

           002152                   .EVEN


002152  047125  047514  042101  MNTBL:  .ASCIZ  /UNLOAD /
002160  000040
002162  042523  045505  020040          .ASCIZ  /SEEK   /
002170  000040
002172  042522  040503  020114          .ASCIZ  /RECAL  /
002200  000040
002202  051104  041526  051114          .ASCIZ  /DRVCLR /
002210  000040
```

```
002212  042522  051514  020105          .ASCIZ  /RELSE  /
002220  000040
002222  043117  051506  052105          .ASCIZ  /OFFSET /
002230  000040
002232  052122  020103  020040          .ASCIZ  /RTC    /
002240  000040
002242  042522  042101  047111          .ASCIZ  /READIN /
002250  000040
002252  040520  045503  020040          .ASCIZ  /PACK   /
002260  000040
002262  042523  051101  044103          .ASCIZ  /SEARCH /
002270  000040
002272  041527  042113  020040          .ASCIZ  /WCKD   /
002300  000040
002302  041527  044113  020104          .ASCIZ  /WCKHD  /
002310  000040
002312  051127  042124  052101          .ASCIZ  /WRTDAT /
002320  000040
002322  051127  044124  020104          .ASCIZ  /WRTHD  /
002330  000040
002332  042122  040504  020124          .ASCIZ  /RDDAT  /
002340  000040
002342  042122  042110  020040          .ASCIZ  /RDHD   /
002350  000040
002352  047516  042516  020040          .ASCIZ  /NONE   /
002360  000040


002362  043101  042524  020122  OFMSG0: .ASCIZ  /AFTER RETRY WITHOUT OFFSET/
002370  042522  051124  020131
002376  044527  044124  052517
002404  020124  043117  051506
002412  052105     000
002415     101  020124  042516  OFMSG1: .ASCIZ  /AT NEGATIVE OFFSET/
002422  040507  044524  042526
002430  047440  043106  042523
002436  000124
002440  052101  050040  051517  OFMSG2: .ASCIZ  /AT POSITIVE OFFSET/
002446  052111  053111  020105
002454  043117  051506  052105
002462     000

        002464                          .EVEN

002464     000                  OFFCOD: .BYTE   0               ;OFFSET CODE TABLE
002465     001                          .BYTE   1               ;NUMBER FOR NEGATIVE OFFSET (DIR = OUT)
002466     000                          .BYTE   0               ;NUMBER FOR POSITIVE OFFSET (DIR = IN)

        002470                  .EVEN

002470  002362                  OFMTBL: .WORD   OFMSG0          ;1ST OFFSET MESSAGE
002472  002415                          .WORD   OFMSG1          ;2ND OFFSET MESSAGE
002474  002440                          .WORD   OFMSG2          ;3RD OFFSET MESSAGE


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

```
                                        .SBTTL  DATA PATTERNS

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

002476  002542                          STNDAT: .WORD   DATA0           ;STANDARD DATA PATTERN POINTER TABLE
002500  002602                                  .WORD   DATA1
002502  002642                                  .WORD   DATA1+40
002504  002702                                  .WORD   DATA1+100
002506  002742                                  .WORD   DATA1+140
002510  003002                                  .WORD   DATA1+200
002512  003042                                  .WORD   DATA1+240
002514  003102                                  .WORD   DATA1+300
002516  003142                                  .WORD   DATA1+340
002520  003202                                  .WORD   DATA1+400
002522  003242                                  .WORD   DATA1+440
002524  003302                                  .WORD   DATA1+500
002526  003342                                  .WORD   DATA1+540
002530  003402                                  .WORD   DATA1+600
002532  003442                                  .WORD   DATA1+640
002534  003502                                  .WORD   DATA1+700
002536  002542                                  .WORD   DATA0           ;ZEROES
002540  003444                                  .WORD   DATA1+642       ;ONES

002542  000000                          DATA0:  .WORD   0               ;DUMMY DATA PATTERN
002544  000000                                  .WORD   0
002546  000000                                  .WORD   0
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  000001                          DATA1:  .WORD   000001  ;STANDARD PATTERN 1
002604  000003                                  .WORD   000003
002606  000007                                  .WORD   000007
002610  000017                                  .WORD   000017
002612  000037                                  .WORD   000037
002614  000077                                  .WORD   000077
002616  000177                                  .WORD   000177
002620  000377                                  .WORD   000377
002622  000777                                  .WORD   000777
002624  001777                                  .WORD   001777
002626  003777                                  .WORD   003777
002630  007777                                  .WORD   007777
002632  017777                                  .WORD   017777
002634  037777                                  .WORD   037777
002636  077777                                  .WORD   077777
```

```
002640  177777                    .WORD   177777
                                  .WORD   177776  ;STANDARD PATTERN 2
002642  177776                    .WORD   177774
002644  177774                    .WORD   177770
002646  177770                    .WORD   177760
002650  177760                    .WORD   177740
002652  177740                    .WORD   177700
002654  177700                    .WORD   177600
002656  177600                    .WORD   177400
002660  177400                    .WORD   177000
002662  177000                    .WORD   176000
002664  176000                    .WORD   174000
002666  174000                    .WORD   170000
002670  170000                    .WORD   160000
002672  160000                    .WORD   140000
002674  140000                    .WORD   100000
002676  100000                    .WORD   000000
002700  000000
                                  .WORD   000000  ;STANDARD PATTERN 3
002702  000000                    .WORD   000000
002704  000000                    .WORD   000000
002706  000000                    .WORD   177777
002710  177777                    .WORD   177777
002712  177777                    .WORD   177777
002714  177777                    .WORD   000000
002716  000000                    .WORD   000000
002720  000000                    .WORD   177777
002722  177777                    .WORD   177777
002724  177777                    .WORD   000000
002726  000000                    .WORD   177777
002730  177777                    .WORD   000000
002732  000000                    .WORD   177777
002734  177777                    .WORD   000000
002736  000000                    .WORD   177777
002740  177777
                                  .WORD   133331  ;STANDARD PATTERN 4
002742  133331                    .WORD   133331
002744  133331                    .WORD   133331
002746  133331                    .WORD   133331
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   052525  ;STANDARD PATTERN 5
003002  052525                    .WORD   052525
003004  052525                    .WORD   052525
003006  052525
```

```
3020  003010  125252                    .WORD   125252
3021  003012  125252                    .WORD   125252
3022  003014  125252                    .WORD   125252
3023  003016  052525                    .WORD   052525
3024  003020  052525                    .WORD   052525
3025  003022  125252                    .WORD   125252
3026  003024  125252                    .WORD   125252
3027  003026  052525                    .WORD   052525
3028  003030  125252                    .WORD   125252
3029  003032  052525                    .WORD   052525
3030  003034  125252                    .WORD   125252
3031  003036  052525                    .WORD   052525
3032  003040  125252                    .WORD   125252
3033
3034  003042  155554                    .WORD   155554  ;STANDARD PATTERN 6
3035  003044  155554                    .WORD   155554
3036  003046  155554                    .WORD   155554
3037  003050  155554                    .WORD   155554
3038  003052  155554                    .WORD   155554
3039  003054  155554                    .WORD   155554
3040  003056  155554                    .WORD   155554
3041  003060  155554                    .WORD   155554
3042  003062  155554                    .WORD   155554
3043  003064  155554                    .WORD   155554
3044  003066  155554                    .WORD   155554
3045  003070  155554                    .WORD   155554
3046  003072  155554                    .WORD   155554
3047  003074  155554                    .WORD   155554
3048  003076  155554                    .WORD   155554
3049  003100  155554                    .WORD   155554
3050
3051  003102  026455                    .WORD   026455  ;STANDARD PATTERN 7
3052  003104  026455                    .WORD   026455
3053  003106  026455                    .WORD   026455
3054  003110  151322                    .WORD   151322
3055  003112  151322                    .WORD   151322
3056  003114  151322                    .WORD   151322
3057  003116  026455                    .WORD   026455
3058  003120  026455                    .WORD   026455
3059  003122  151322                    .WORD   151322
3060  003124  151322                    .WORD   151322
3061  003126  026455                    .WORD   026455
3062  003130  151322                    .WORD   151322
3063  003132  026455                    .WORD   026455
3064  003134  151322                    .WORD   151322
3065  003136  026455                    .WORD   026455
3066  003140  151322                    .WORD   151322
3067
3068  003142  066666                    .WORD   066666  ;STANDARD PATTERN 8
3069  003144  066666                    .WORD   066666
3070  003146  066666                    .WORD   066666
3071  003150  066666                    .WORD   066666
3072  003152  066666                    .WORD   066666
3073  003154  066666                    .WORD   066666
3074  003156  066666                    .WORD   066666
3075  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  000001                    .WORD   000001  ;STANDARD PATTERN 9
003204  000002                    .WORD   000002
003206  000004                    .WORD   000004
003210  000010                    .WORD   000010
003212  000020                    .WORD   000020
003214  000040                    .WORD   000040
003216  000100                    .WORD   000100
003220  000200                    .WORD   000200
003222  000400                    .WORD   000400
003224  001000                    .WORD   001000
003226  002000                    .WORD   002000
003230  004000                    .WORD   004000
003232  010000                    .WORD   010000
003234  020000                    .WORD   020000
003236  040000                    .WORD   040000
003240  100000                    .WORD   100000

003242  177776                    .WORD   177776  ;STANDARD PATTERN 10
003244  177775                    .WORD   177775
003246  177773                    .WORD   177773
003250  177767                    .WORD   177767
003252  177757                    .WORD   177757
003254  177737                    .WORD   177737
003256  177677                    .WORD   177677
003260  177577                    .WORD   177577
003262  177377                    .WORD   177377
003264  176777                    .WORD   176777
003266  175777                    .WORD   175777
003270  173777                    .WORD   173777
003272  167777                    .WORD   167777
003274  157777                    .WORD   157777
003276  137777                    .WORD   137777
003300  077777                    .WORD   077777

003302  172666                    .WORD   172666  ;STANDARD PATTERN 11
003304  155555                    .WORD   155555
003306  172666                    .WORD   172666
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  077777                    .WORD   077777  ;STANDARD PATTERN 12
003344  137777                    .WORD   137777
003346  157777                    .WORD   157777
003350  167777                    .WORD   167777
003352  173777                    .WORD   173777
003354  175777                    .WORD   175777
003356  176777                    .WORD   176777
003360  177377                    .WORD   177377
003362  177577                    .WORD   177577
003364  177677                    .WORD   177677
003366  177737                    .WORD   177737
003370  177757                    .WORD   177757
003372  177767                    .WORD   177767
003374  177773                    .WORD   177773
003376  177775                    .WORD   177775
003400  177776                    .WORD   177776

003402  153333                    .WORD   153333  ;STANDARD PATTERN 13
003404  066667                    .WORD   066667
003406  153333                    .WORD   153333
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  000000                    .WORD   000000  ;STANDARD PATTERN 14
003444  177777                    .WORD   177777
003446  177777                    .WORD   177777
003450  177777                    .WORD   177777
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  ;STANDARD PATTERN 15
```

```
3188  003504  000000                          .WORD   000000
3189  003506  000000                          .WORD   000000
3190  003510  000000                          .WORD   000000
3191  003512  000000                          .WORD   000000
3192  003514  000000                          .WORD   000000
3193  003516  000000                          .WORD   000000
3194  003520  000000                          .WORD   000000
3195  003522  000000                          .WORD   000000
3196  003524  000000                          .WORD   000000
3197  003526  000000                          .WORD   000000
3198  003530  000000                          .WORD   000000
3199  003532  000000                          .WORD   000000
3200  003534  000000                          .WORD   000000
3201  003536  000000                          .WORD   000000
3202  003540  000000                          .WORD   000000
3203
```

```
                                .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


003542                          $ERRTB:
                                ;ERROR 1

                                ;ERROR 1

003542  045622                          EM1                     ;RH70 INTERRUPT OCCURRED (RMAS = 0)
003544  050241                          DH1
003546  050672                          DT1
003550  000000                          0

                                ;ERROR 2

003552  045671                          EM2                     ;UNEXPECTED ATTENTION OCCURRED
003554  050246                          DH2
003556  050676                          DT2
003560  000000                          0

                                ;ERROR 3

003562  045727                          EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
003564  050322                          DH3
003566  050714                          DT3
003570  000000                          0

                                ;ERROR 4

003572  045765                          EM4                     ;MASSBUS PARITY ERROR (PAR=1)
003574  050350                          DH4
003576  050724                          DT4
003600  000000                          0

                                ;ERROR 5

003602  046022                          EM5                     ;ADDRESS PLUG BIT CHANGED
003604  050246                          DH2
003606  050676                          DT2
003610  000000                          0

                                ;ERROR 6

003612  046056                          EM6                     ;RH11 DIDN'T RESPOND TO ADDRESSING
003614  050407                          DH6
```

```
003616  050736                          DT6
003620  000000                          0

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

                                .SBTTL  SETUP AND INITIALIZATION ROUTINE

                                ;       START ADDRESS = 200
                                ;       ADDRESS TO CHANGE RH11 UNIBUS ADDRESS = 204

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

003622  012737  177777  001356  START:  MOV     #-1,CHGADR      ;SET RH11 ADDRESS CHANGE FLAG
003630  000407                          BR      START2          ;START THE PROGRAM
003632  012737  000400  001356  START1: MOV     #400,CHGADR     ;CLEAR THE RH11 ADDRESS CHANGE FLAG
                                ;;*****************************************************************
003640  000240                  TST1:   NOP
003642  012737  000001  001210          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
003650  000005                  START2: RESET                   ;CLEAR THE BUS
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
003652  012706  001114                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
003656  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
003660  022706  001154                  CMP     #SWR,R6 ;;DONE?
003664  001374                          BNE     .-6             ;;LOOP BACK IF NO
003666  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
003672  012737  031006  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
003700  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
003706  012737  034062  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
003714  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
003722  012737  032056  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
003730  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
003736  012737  176543  033446          MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
003744  012737  123456  033450          MOV     #123456,$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.
003752  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
003756  012737  004012  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
003764  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
003772  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
004000  022777  177777  175146          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
004006  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
004010  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
004012  012716  004020          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
004016  000002                          RTI
004020  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
004026  012737  000174  001156          MOV     #DISPREG,DISPLAY
004034  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR .

004040  005037  001212                  CLR     $PASS           ;;CLEAR PASS COUNT
004044  132737  000200  001225          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
004052  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
004054  012737  001226  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
004062                          67$:
```

```
004052  012737  000240  000032          MOV     #240,@#EMTVEC+2     ;CHANGE EMT PRIORITY TO 4
004060  012737  000240  000036          MOV     #240,@#TRAPVEC+2    ;CHANGE TRAP PRIORITY TO 4
004066  005237  177777                  INC     #-1                 ;FIRST START ?
004072  001003                          BNE     1$                  ;BR IF NOT
004074  104401  057150                  TYPE    ,TITLE              ;MOVE AND MANDEC NUMBER
004110  005737  000042                  TST     42                  ;AUTO ACCEPT OR CHAIN MODE ?
004114  001006                          BNE     1$                  ;BR IF EITHER
004116  105737  000011  000041          CMPB    #11,41              ;LOADED FROM AN RM03 ?
004124  001002                          BNE     1$                  ;BR IF NOT
004126  104401  057233                  TYPE    ,LOADRV             ;INSTRUCT THE OPERATOR ON HOW TO TEST DRIVE 0
004132  004737  030326          1$:     JSR     PC,$TKINT           ;TURN ON THE KEYBOARD INTERRUPT
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
004136  005737  000042                  TST     @#42                ;;ARE WE RUNNING UNDER XXDP/ACT?
004142  001012                          BNE     68$                 ;;BRANCH IF YES
004144  123727  001224  000001          CMPB    $ENV,#1             ;;ARE WE RUNNING UNDER APT?
004152  001406                          BEQ     68$                 ;;BRANCH IF YES
004154  023727  001154  000176          CMP     SWR,#SWREG          ;;SOFTWARE SWITCH REG SELECTED?
004162  001005                          BNE     69$                 ;;BRANCH IF NO
004164  104406                          GTSWR                       ;;GET SOFT-SWR SETTINGS
004166  000403                          BR      69$
004170  112737  000001  001150  68$:    MOVB    #1,$AUTOB           ;;SET AUTO-MODE INDICATOR
004176                          69$:
004176  105737  001224                  TSTB    $ENV                ;RUN UNDER APT MODE
004202  001423                          BEQ     4$                  ;NO,DONOT BOTHER
004204  105737  001254                  TSTB    $VECT1              ;NEW VECTOR ?
004210  001403                          BEQ     .+10                ;NOT LOAD IF = 0
004212  113737  001254  001272          MOVB    $VECT1,$RMVEC       ;NEW VECTOR
004220  005737  001260                  TST     $BASE               ;NEW BASE ADDRESS ?
004224  001403                          BEQ     .+10                ;NO
004226  013737  001260  001270          MOV     $BASE,$RMADR        ;NEW BASE ADDRESS
004234  013737  001270  034316          MOV     $RMADR,RMADR        ;LOAD ADDRESS INTO DRIVER
004242  013737  001272  034320          MOV     $RMVEC,RMVEC        ;LOAD VECTOR INTO DRIVER
004250  000420                          BR      2$
004252  105737  001150          4$:     TSTB    $AUTOB              ;AUTO MODE ?
004256  001015                          BNE     2$                  ;YES
004260  004737  055676                  JSR     PC,BUSADR           ;CHECK RH11 BUS ADDRESS
004264  013737  001270  034316          MOV     $RMADR,RMADR        ;RH11 ADDRESS
004272  013737  001272  034320          MOV     $RMVEC,RMVEC        ;RH11 VECTOR ADDRESS
004300  005227  177777                  INC     #-1                 ;FIRST START ?
004304  001002                          BNE     2$                  ;BRANCH IF NOT
004306  004737  055410                  JSR     PC,OPRDAT           ;GET THE DATE AND OPERATOR ID
004312  005037  001314          2$:     CLR     STATIN              ;CLEAR PERFORMANCE SUMMARY TYPEOUT FLAG
004316  012705  001522                  MOV     #ORDERQ,R5          ;START OF AREA TO CLEAR
004322  005025                  3$:     CLR     (R5)+
004324  022705  002102                  CMP     #BLKADR,R5          ;LOOK FOR END OF CLEAR AREA
004330  001374                          BNE     3$                  ;BR IF NOT FINISHED
004332  012706  001100                  MOV     #STACK,SP           ;SETUP THE STACK POINTER
004336  005037  177776                  CLR     PS                  ;CLEAR THE PROCESSOR STATUS WORD
004342  013737  001312  001372          MOV     HZ,SIXTEE           ;1/60 TH OR 1/50 TH SECOND COUNTER VALUE
004350  005037  001364                  CLR     HOUR                ;CLEAR THE HOUR'S COUNTER
004354  005037  001366                  CLR     MINUTE              ;CLEAR THE MINUTE'S COUNTER
004360  005037  001370                  CLR     SECOND              ;CLEAR THE SECOND'S COUNTER
004364  005037  001470                  CLR     INTRVL+2            ;CLEAR INTERVAL COUNTER
004370  005037  001316                  CLR     PACK                ;CLEAR THE 'R' OR 'W' COMMAND FLAG
004374  005037  001360                  CLR     CFLAG               ;CLEAR THE 'CONTROL C' FLAG
004400  042737  170000  001464          BIC     #170000,MAXER       ;MAKE SURE ERROR LIMITS ARE NOT TOO HIGH
```

```
                                        ;ROUTINE TO DETERMINE BUFFER AREA SIZE

004406  005227  177777          SIZMEM: INC     #-1             ;SEE IF TIME TO SIZE MEMORY
004412  001005                          BNE     1$              ;BR IF NOT
004414  004737  055600                  JSR     PC,$SIZE        ;SEE HOW MUCH MEMORY ON SYSTEM
004420  013737  055674  001354          MOV     $LSTAD,LSTAD    ;SAVE THE LAST ADDRESS
004426  012737  000001  001700  1$:     MOV     #1,BUFTBL       ;LOAD NUMBER OF BUFFERS
004434  012737  057150  001702          MOV     #ENDPGM,BUFTBL+2  ;STARTING ADDRESS OF BUFFER
004442  013737  001354  001704          MOV     LSTAD,BUFTBL+4  ;LAST ADDR TO BUFFER ALLOCATION TABLE
004450  023727  001354  160000          CMP     LSTAD,#160000   ;OVER 28K ?
004456  101403                          BLOS    2$              ;NO
004460  012737  160000  001704          MOV     #160000,BUFTBL+4        ;XXDP MAX MEMORY 28K
004466  162737  057150  001704  2$:     SUB     #ENDPGM,BUFTBL+4  ;SUBTRACT PROGRAM SPACE
004474  000241                          CLC                     ;CLEAR THE 'C' BIT
004476  006037  001704                  ROR     BUFTBL+4        ;CONVERT TO WORD COUNT
004502  162737  000144  001704          SUB     #100.,BUFTBL+4  ;SAVE ROOM FOR THE 'ABS' LOADER
004510  005737  000042                  TST     42              ;LOAD FROM XXDP OR OTHER MONITOR ?
004514  001403                          BEQ     3$              ;BR IF LOADED BY PAPER TAPE
004516  162737  002570  001704          SUB     #1400.,BUFTBL+4  ;SUBTRACT 'XXDP' LOADER SIZE
004524  005737  001462          3$:     TST     MAXDL           ;VALUE IN 'MAXDL' ?
004530  001003                          BNE     4$              ;BR IF VALUE IS
004532  012737  020000  001462          MOV     #8192.,MAXDL    ;ASSUME FULL TRACK   MAXIMUM
004540  023737  001462  001704  4$:     CMP     MAXDL,BUFTBL+4  ;IS THAT TOO LARGE ?
004546  103403                          BLO     5$              ;BR IF NOT
004550  013737  001704  001462          MOV     BUFTBL+4,MAXDL  ;USE MAX AVAIL MEMORY AS MAX BUFFER SIZE
004556  013737  001704  054354  5$:     MOV     BUFTBL+4,PARLST+2  ;VALUE FOR THE PARAMETER TABLE

                                        ;SEE IF THE OPERATOR WANTS TO CHANGE ANY PARAMETERS

004564  105737  001150          LKPAR:  TSTB    $AUTOB          ;'XXDP' CHAIN MODE OR 'ACT11' OPERATION ?
004570  001027                          BNE     SETVEC          ;BR IF YES
004572  105737  001224                  TSTB    $ENV            ;APT STAND ALONE MODE ?
004576  001024                          BNE     SETVEC          ;NO
004600  104401  001201                  TYPE    ,$CRLF
004604  104401  054450                  TYPE    ,ASKPAR         ;ASK FOR PARMETERS
004610  104411                          RDLIN                   ;READ THE ENTRY
004612  012605                          MOV     (SP)+,R5        ;ADDRESS OF ENTRY TO R5
004614  122715  000131                  CMPB    #'Y,(R5)        ;WAS ENTRY A 'Y' (YES)
004620  001013                          BNE     SETVEC          ;BR IF NOT 'Y'
004622  012703  054352          ENTPR:  MOV     #PARLST,R3      ;PARAMETER TABLE ADDRESS
004626  004737  026420                  JSR     PC,PARENT       ;GET THE PARAMETER ENTRY
004632  023727  001462  000004          CMP     MAXDL,#4        ;IS THE 'MAXDL' VALUE OK ?
004640  103003                          BHIS    SETVEC          ;BR IF IT IS
004642  012737  000004  001462          MOV     #4,MAXDL        ;SET 'MAXDL' TO THE MINIMUM VALUE

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

004650  004737  022412          SETVEC: JSR     PC,CKCLK        ;START THE CLOCK
004654  004737  034334                  JSR     PC,RMINIT       ;INITIALIZE THE RM03 DRIVER
004660  012737  177777  034256          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
004666  062727  177777  000000          ADD     #-1,#0          ;CHECK FOR FIRST START
004674  103004                          BCC     11$             ;BR IF FIRST START
004676  032777  000004  174250          BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
004704  001061                          BNE     10$             ;BR IF NOT
```

```
004706  005004                  11$:    CLR     R4              ;DRIVE TABLE POINTER
004710  104401  001201                  TYPE    ,$CRLF          ;CR-LF
004714  104401  053135                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
004720                          1$:
004720  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
004722  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
004724     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
004725     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
004726  104401  052714                  TYPE    ,LINSP          ;SPACES
004732  105764  034170                  TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
004736  100416                          BMI     4$              ;BR IF UNSAFE
004740  001020                          BNE     5$              ;BR IF ONLINE
004742  105764  034200                  TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
004746  001404                          BEQ     2$              ;BR IF NONEXISTENT
004750  100006                          BPL     3$              ;BR IF OFFLINE
004752  104401  053054          12$:    TYPE    ,NOTRM          ;DRIVE NOT AN RM03
004756  000426                          BR      9$              ;CHECK NEXT DRIVE
004760  104401  053071          2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
004764  000423                          BR      9$              ;CHECK NEXT DRIVE
004766  104401  052763          3$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
004772  000405                          BR      6$              ;PRINT DRIVE TYPE
004774  104401  053125          4$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
005000  000402                          BR      6$              ;PRINT DRIVE TYPE
005002  104401  052774          5$:     TYPE    ,UNTON          ;DRIVE ONLINE
005006  104401  052716          6$:     TYPE    ,LINSP          ;SPACES
005012  012737  053155  005032          MOV     #RM03A,8$       ;ADDRESS OF RM03 MESSAGE
005020  132764  000004  034200          BITB    #BIT02,DRVTYP(R4)   ;RM03 ?
005026  001751                          BEQ     12$             ;BR IF NO TYPE NOT RM03 MSG
005030  104401                  7$:     TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
005032  000000                  8$:     .WORD   0               ;MESSAGE ADDRESS HERE
005034  104401  001201          9$:     TYPE    ,$CRLF          ;CR-LF
005040  005204                          INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
005042  020427  000010                  CMP     R4,#8.          ;FINISHED ?
005046  001324                          BNE     1$              ;BR IF NOT
005050  104401  001201          10$:    TYPE    ,$CRLF          ;CR-LF
005054  005037  177776                  CLR     PS              ;SET PRIORITY BACK TO '0'
005060  000137  005064                  JMP     MONTR           ;CHECK FOR 'XXDP' OR 'ACT11' MONITOR

                                ;SETUP IF 'XXDP' OR 'ACT11' OPERATION

005064  005737  000042          MONTR:  TST     42              ;'XXDP' CHAIN MODE OR 'ACT11' AUTO ACCEPT
005070  001405                          BEQ     1$              ;BR IF NEITHER
005072  005737  001356                  TST     CHGADR          ;200 START ?
005076  003002                          BGT     1$              ;YES
005100  004737  024324                  JSR     PC,ASGN2        ;ASSIGN DRIVES
005104  005227  177777          1$:     INC     #-1             ;FIRST START ?
005110  001011                          BNE     2$              ;BR IF NOT
005112  105737  000041                  TSTB    @#41            ;LOADED FROM PAPER TAPE ?
005116  001406                          BEQ     2$              ;BR IF YES
005120  023727  001354  100000          CMP     LSTAD,#100000   ;MORE THAN 16K ON THE SYSTEM ?
005126  103002                          BHIS    2$              ;BR IF YES
005130  104401  057432                  TYPE    ,NOLOAD         ;TELL THE OPERATOR THAT THE 'XXDP' LOADER
                                                                ;WILL BE OVERWRITTEN
005134  004737  030326          2$:     JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD INTERRUPT HANDLER
                                ;       FORCE TO   TEST PACK FOR 200 START
```

```
005140  105737  001150          FOWT1:  TSTB    $AUTOB          ;AUTO MODE ?
005144  001412                          BEQ     1$              ;NO
005146  012737  000001  001500          MOV     #1,RATIO        ;SPEED UP TEST
005154  012737  077777  001450          MOV     #77777,ENDCON   ;SET LSW OF ENDING VALUE
005162  012737  000027  001452          MOV     #27,ENDCON+2    ;SET MSW OF ENDING VALUE
005170  000403                          BR      FOWT2           ;START TO WRITE AND TEST
005172  005737  001356          1$:     TST     CHGADR          ;START AT 200 ?
005176  003461                          BLE     FOWT3           ;NO
005200  005001                  FOWT2:  CLR     R1              ;DRIVE #
005202  005002                          CLR     R2              ;AVAIL TABLE INDEX
005204  005003                          CLR     R3              ;DRIVE# X 2
005206  122737  000024  000041          CMPB    #24,41          ;LOAD FROM DRIVE RM03
005214  001002                          BNE     1$              ;NO
005216  005201                          INC     R1              ;START FROM DRIVE 1
005220  005723                          TST     (R3)+           ;ADJUST INDEX
005222  105761  034170          1$:     TSTB    DRVSTA(R1)      ;DRIVE ON LINE ?
005226  003434                          BLE     2$              ;NO
005230  016300  002102                  MOV     BLKADR(R3),R0   ;LOAD DPB ADDRESS
005234  004737  025212                  JSR     PC,CLRDPB       ;CLEAR DPB BLOCK
005240  004537  025764                  JSR     R5,GETADR       ;RETRIEVE BAD SPO. FILE
005244  000425                          BR      2$              ;ERROR RET FROM GETADR RT.
005246  010062  001570                  MOV     R0,NEWUNT(R2)   ;LOAD DPB ADDRESS TO ABAIL QUEUE
005252  105060  000026                  CLRB    $PACK(R0)       ;TEST PACK FLAG
005256  013760  001446  000106          MOV     CYLIMT,MAXCYL(R0) ;UP CYLINDER LIMIT
005264  013760  001444  000112          MOV     TRKLMT,MAXTRK(R0) ;UPPER TRACK LIMIT
005272  013760  001442  000116          MOV     SECLMT,MAXSEC(R0) ;UPPER SECTOR LIMIT
005300  005060  000110                  CLR     MINCYL(R0)      ;CLEAR LOWER LIMIT
005304  005060  000114                  CLR     MINTRK(R0)      ;CLEAR LOWER LIMIT
005310  005060  000120                  CLR     MINSEC(R0)      ;CLEAR LOWER LIMIT
005314  004737  017474                  JSR     PC,INPTPK       ;SET UP PARAMETERS
005320  022322                  2$:     CMP     (R3)+,(R2)+     ;INCREMENT INDEX
005322  005201                          INC     R1              ;NEXT DRIVE
005324  022701  000007                  CMP     #7,R1           ;ALL DRIVE ASSIGN ?
005330  002334                          BGE     1$              ;NO
005332  005037  001316                  CLR     PACK            ;TEST PACK FLAG
005336  000137  005524                  JMP     MAIN1           ;JUMP TO WAIT PARAMETER AND BUFFER LOOP
005342  104401  054120          FOWT3:  TYPE    ,INTDON         ;TYPE 'INITIALIZE COMPLETE'
005346  000137  005524                  JMP     MAIN1           ;START THE PROGRAM
                                ;;$DOAGN:       JMP     START           ;START AGAIN

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

                                .SBTTL  MAIN PROGRAM

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

005352  005737  001522          MAIN:   TST     ORDERQ          ;DON'T DEASSIGN ANY DRIVE IF SYSTEM NOT IDLE
005356  001402                          BEQ     1$              ;LET ALL DRIVE FINISH WITH COMMAND
005360  000137  006006                  JMP     IDLE
005364  005737  001360          1$:     TST     CFLAG           ;KEYBOARD INTERRUPTTED ?
005370  001402                          BEQ     2$              ;BRANCH IF NOT
005372  004737  023710                  JSR     PC,KSR          ;SERVICE THE KEYBOARD
005376                          2$:
005376  012703  000010          MAINDA: MOV     #8.,R3          ;DRIVE COUNTER
005402  012705  001546                  MOV     #DUNIT,R5       ;ADDRESS OF 'DROP DRIVE' TABLE
005406  005715                  1$:     TST     (R5)            ;SEE IF ENTRY AT PRESENT POSITION
```

```
005410  001011                          BNE     3$              ;BR IF THERE IS ONE
005412  062705  000002          2$:     ADD     #2,R5           ;INCREMENT TO NEXT TABLE POSITION
005416  005303                          DEC     R3              ;DECREMENT DRIVE COUNTER
005420  001372                          BNE     1$              ;BR IF MORE TO CHECK
005422  105737  001544                  TSTB    ASNLST          ;ANY DRIVES ACTIVE ?
005426  001036                          BNE     MAIN1           ;BR IF YES
005430  000137  027250                  JMP     $GET42          ;CHECK FOR MONITOR RETURN
005434  012701  001612          3$:     MOV     #AVAIL,R1       ;ADDRESS OF 'AVAILABLE DRIVES' TABLE
005440  005711                  4$:     TST     (R1)            ;SEE IF AT END OF TABLE
005442  001405                          BEQ     5$              ;BR IF AT END: GO CHECK 'WAIT' TABLE
005444  021115                          CMP     (R1),(R5)       ;IS DRIVE IN 'AVAIL' THE ONE TO BE DROPPED
005446  001414                          BEQ     7$              ;BR IF YES
005450  062701  000002                  ADD     #2,R1           ;INCREMENT 'AVAIL' TABLE ADDRESS
005454  000771                          BR      4$              ;CONTINUE LOOKING
005456  012701  001634          5$:     MOV     #WAIT,R1        ;MOVE THE ADDRESS OF THE BUFFER WAIT TABLE
005462  005711                  6$:     TST     (R1)            ;AT THE END OF THE 'WAIT' TABLE ?
005464  001752                          BEQ     2$              ;BR IF YES: SEE IF ANY MORE 'DROP' REQUESTS
005466  021115                          CMP     (R1),(R5)       ;DRIVE IN THE 'WAIT' TABLE ?
005470  001403                          BEQ     7$              ;BR IF IT IS
005472  062701  000002                  ADD     #2,R1           ;INCREMENT 'WAIT' TABLE ADDRESS
005476  000771                          BR      6$              ;CONTINUE LOOK THROUGH THE 'WAIT' TABLE
005500  011100                  7$:     MOV     (R1),R0         ;PUT THE DRIVE'S BLOCK ADDRESS IN R0
005502  104401  053427                  TYPE    ,DEASSG         ;TYPE 'DRIVE DEASSIGNED'
005506  004737  022702                  JSR     PC,TYPEST       ;TYPE THE DRIVE'S PERFORMANCE SUMMARY
005512  005015                          CLR     (R5)            ;CLEAR THE 'DROP DRIVE' TABLE ENTRY
005514  005011                          CLR     (R1)            ;REMOVE THE DRIVE FROM THE 'AVAIL' OR 'WAIT' TABLE
005516  004737  017456                  JSR     PC,CMPRES       ;COMPRESS THE RESPECTIVE TABLE
005522  000733                          BR      2$              ;SEE IF ANY MORE DRIVES

                                ;LOOK FOR DRIVES TO BE ASSIGNED

005524  012703  000010          MAIN1:  MOV     #8.,R3          ;DRIVE COUNT
005530  005002                          CLR     R2              ;'AVAIL' INDEX
005532  005004                          CLR     R4              ;ASSIGN LIST INDEX
005534  005005                          CLR     R5              ;NEW DRIVE INDEX
005536  005765  001570          1$:     TST     NEWUNT(R5)      ;NEW DRIVE IN THIS POSITION
005542  001006                          BNE     3$              ;BR IF THERE IS
005544  062705  000002          2$:     ADD     #2,R5           ;INCREMENT R5
005550  005204                          INC     R4              ;INCREMENT ASSIGN INDEX
005552  005303                          DEC     R3              ;DECREMENT DRIVE COUNT
005554  001370                          BNE     1$              ;BR IF MORE DRIVES
005556  000430                          BR      MAIN2           ;START OPERATIONS FOR THE AVAILABLE DRIVES
005560  104401  052755          3$:     TYPE    ,UNTMSG         ;'DRIVE'
005564  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
005566  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
005570     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
005571     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
005572  104401  053477                  TYPE    ,ASGND          ;'STARTED'
005576  005762  001612          4$:     TST     AVAIL(R2)       ;AT END OF AVAILABLE TABLE
005602  001403                          BEQ     5$              ;BR IF YES
005604  062702  000002                  ADD     #2,R2           ;INCREMENT AVAILABLE TABLE INDEX
005610  000772                          BR      4$              ;CONTINUE LOOKING FOR END OF TABLE
005612  016562  001570  001612  5$:     MOV     NEWUNT(R5),AVAIL(R2)  ;MOVE ADDR OF DRIVE INTO AVAIL LST
005620  005065  001570                  CLR     NEWUNT(R5)      ;TAKE DRIVE OUT OF NEW DRIVE TABLE
005624  156437  034304  001544          BISB    ATABIT(R4),ASNLST ;SET DRIVE ASSIGNED INDICATOR
```

```
005632  062702  000002                  ADD     #2,R2           ;INCREMENT AVAILABLE TABLE POINTER
005636  000742                          BR      2$              ;LOOK FOR MORE DRIVES

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

005640  005737  001522          MAIN2:  TST     ORDERQ          ;OUTSTANDING BUFFER REQUESTS
005644  001060                          BNE     IDLE            ;BR IF THERE ARE
005646  005002                          CLR     R2              ;CLEAR DRIVE TABLE POINTER
005650  005762  001612          1$:     TST     AVAIL(R2)       ;ANY DRIVES WAITING FOR PARAMETERS
005654  001002                          BNE     .+6             ;BRANCH IF ANY
005656  000137  006006                  JMP     IDLE            ;BRANCH IF NONE
005662  016200  001612                  MOV     AVAIL(R2),R0    ;CONTROL BLOCK ADDR IN R0
005666  005760  000104                  TST     $NEXT(R0)       ;PARAMETERS BEEN SELECTED ?
005672  001010                          BNE     6$              ;BR IF THEY HAVE
005674  105760  000026                  TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR THE DRIVE ?
005700  001403                          BEQ     5$              ;BR IF NOT
005702  004737  017474                  JSR     PC,WRTPK        ;GET DATA PACK PARAMETERS
005706  000404                          BR      7$              ;GET THE BUFFER
005710  004737  016210          5$:     JSR     PC,SELPAR       ;SELECT THE PARAMETERS
005714  004737  017164          6$:     JSR     PC,GETPAR       ;LOAD NEW PARAMETERS
005720  005046                  7$:     CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
005722  004737  015420                  JSR     PC,GETBUF       ;GET BUFFER
005726  012660  000006                  MOV     (SP)+,$BUF(R0)  ;MOVE BUFFER ADDR TO DPB
005732  001414                          BEQ     8$              ;BR IF '0' ADDR (NO BUFFER)
005734  004737  016004                  JSR     PC,FILBUF       ;FILL THE BUFFER
005740  005060  000072                  CLR     $FAIR(R0)       ;CLEAR THE 'FAIRNESS' COUNT
005744  004737  016132                  JSR     PC,GODRIV       ;PUT CURRENT DPB IN DRIVER
005750  012705  001522                  MOV     #ORDERQ,R5      ;ADDRESS OF ORDER QUEUE IN R5
005754  005725                          TST     (R5)+           ;END OF QUEUE ?
005756  001376                          BNE     .-2             ;BR IF NOT
005760  010045                          MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO QUEUE
005762  000403                          BR      10$             ;CONTINUE LOOKING
005764  062702  000002          8$:     ADD     #2,R2           ;INCREMENT INDEX
005770  000727                          BR      1$              ;BRANCH BACK TO FIRE NEXT DRIVE
005772  012701  001612          10$:    MOV     #AVAIL,R1       ;'AVAILABLE' TABLE ADDRESS
005776  060201                          ADD     R2,R1           ;FORM ADDRESS OF LAST ENTRY
006000  004737  017456                  JSR     PC,CMPRES       ;COMPRESS THE TABLE
006004  000721                          BR      1$              ;CONTINUE LOOKING

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


                                ;WAIT FOR AN ORDER TO FINISH

006006  012701  001522          IDLE:   MOV     #ORDERQ,R1      ;ADDRESS OF THE ORDER QUEUE IN R1
006012  012100                  1$:     MOV     (R1)+,R0        ;PUT BLOCK ADDRESS INTO R0
006014  001433                          BEQ     IDLE1           ;BR IF END OF QUEUE
006016  005760  000016                  TST     $STATUS(R0)     ;SEE IF DRIVE FINISHED
006022  001773                          BEQ     1$              ;BR IF DRIVE NOT FINISHED
006024  162701  000002                  SUB     #2,R1           ;CORRECT THE QUEUE POINTER
006030  010146                          MOV     R1,-(SP)        ;SAVE THE QUEUE ADDRESS
006032  004737  015262                  JSR     PC,STATIS       ;ACCUMULATE STATISTICS FOR DRIVE IN R0
006036  000240                          NOP                     ;DEBUGGING AID
006040  004737  006260                  JSR     PC,PROCES       ;PROCESS END OF ORDER
006044  005037  001362                  CLR     BADSEC          ;CLEAR THE BAD TRK/SEC ERROR INDICATOR
```

```
006050  004737  026674                  JSR     PC,ABNRML       ;SEE IF ANY DRIVES HAVE TOO MANY ERRORS
006054  004737  026722                  JSR     PC,EOP          ;SEE IF ANY DRIVE HAS XFERED 3X10↑9 BITS
006060  012601                          MOV     (SP)+,R1        ;RESTORE THE ORDER TABLE INDEX
006062  012705  001612                  MOV     #AVAIL,R5       ;FIND THE END OF THE 'AVAILABLE' TABLE
006066  005725                  2$:     TST     (R5)+           ;END OF THE TABLE ?
006070  001376                          BNE     2$              ;BR IF NOT AT END OF LIST
006072  011145                          MOV     (R1),-(R5)      ;MOVE THE BLOCK ADDRESS INTO THE TABLE
006074  004737  017456                  JSR     PC,CMPRES       ;COMPRESS THE ORDER QUEUE
006100  004737  015554                  JSR     PC,RELBUF       ;RESTORE BUFFER
006104                          IDLE1:
006104  032777  000004  173042  1$:     BIT     #SW02,@SWR      ;TYPE PERFORMANCE SUMMARY
006112  001007                          BNE     2$              ;BR IF NOT
006114  005737  001314                  TST     STATIN          ;TIME TO TYPE THE PERFORMANCE SUMMARY ?
006120  001404                          BEQ     2$              ;BR IF NOT
006122  005037  001314                  CLR     STATIN          ;CLEAR THE INDICATOR
006126  004737  022614                  JSR     PC,STATPR       ;TYPE THE SUMMARY
006132  000137  005352          2$:     JMP     MAIN            ;CONTINUE THE LOOP

                                ;SETUP TO REFORMAT AN ERROR SECTOR

006136  032777  000001  173010  REFMT:  BIT     #SW0,@SWR       ;READ ONLY SWITCH SET ?
006144  001044                          BNE     REFMTX          ;BR IF IT IS
006146  032777  000200  173000          BIT     #SW7,@SWR       ;SWITCH 7 SET ?
006154  001040                          BNE     REFMTX          ;BR IF IT IS
006156  005737  001474                  TST     FORMAT          ;WRITE HEADER & DATA ORDERS ALLOWED ?
006162  001435                          BEQ     REFMTX          ;BR IF NOT
006164  022760  001465  000270          CMP     #821.,$RMDC(R0) ;LEGAL VALUE ?
006172  101431                          BLOS    REFMTX          ;NO, NOT FORMAT
006174  004737  022262                  JSR     PC,READDR       ;GET CORRECTED SECTOR-TRACK ADDRESSES
006200  012660  000100                  MOV     (SP)+,$NCYL(R0)         ;CYLINDER
006204  112660  000077                  MOVB    (SP)+,$NTRK(R0) ;TRACK ADDR TO DPB
006210  112660  000076                  MOVB    (SP)+,$NSEC(R0) ;SECTOR ADDR TO DPB
006214  012760  000402  000102          MOV     #258.,$NWRDL(R0)  ;WORD COUNT FOR FORMAT
006222  112760  000003  000074  1$:     MOVB    #3,$NCODE(R0)   ;COMMAND CODE
006230  004537  016762                  JSR     R5,CHKADR       ;AVOID REFORMAT BAD SPOT
006234  000401                          BR      2$              ;BRANCH IF NOT ON BAD SPOT
006236  000407                          BR      REFMTX          ;BRANCH IF ON BAD SPOT
006240  004737  017122          2$:     JSR     PC,GETPAT       ;GET A PATTERN
006244  110560  000075                  MOVB    R5,$NPATC(R0)   ;PATTERN CODE TO CONTROL BLOCK
006250  012760  177777  000104          MOV     #-1,$NEXT(R0)   ;SET PARAMETERS SELECTED INDICATOR
006256  000207                  REFMTX: RTS     PC              ;RETURN

                                ;PROCESS THE ORDER TERMINATION

006260  111037  001344          PROCES: MOVB    (R0),UNIT       ;DRIVE NUMBER FOR ANY ERROR MESSAGES
006264  005760  000016                  TST     $STATUS(R0)     ;SEE IF DRIVER SIGNALED AN ERROR
006270  100427                          BMI     ERPROC          ;BR IF ERROR
006272  032760  100000  000234          BIT     #BIT15,$RMCS1(R0)  ;SEE IF 'SC' SET
006300  001410                          BEQ     1$              ;BR IF NOT SET
006302  032760  040000  000234          BIT     #BIT14,$RMCS1(R0)  ;SEE IF 'TRE' SET
006310  001017                          BNE     ERPROC          ;BR IF SET
006312  032760  040000  000246          BIT     #BIT14,$RMDS(R0)  ;SEE IF 'ERR' SET
006320  001013                          BNE     ERPROC          ;BR IF SET
006322  004737  012440          1$:     JSR     PC,CKERR        ;NO ERROR, CHECK ERROR BITS ANYWAY
006326  004737  012532                  JSR     PC,CKBUS        ;NO ERROR, CHECK BUS ADDR & WC
006332  032777  000002  172614          BIT     #SW01,@SWR      ;DATA COMPARE ?
```

```
006340  001002                          BNE     2$              ;BR IF NOT
006342  004737  012616                  JSR     PC,CMPAR        ;NO ERROR, COMPARE DATA
006346  000207                  2$:     RTS     PC              ;RETURN

                                ;ORDER TERMINATED WITH AN ERROR - PROCESS THE ERROR

006350  032760  000200  000016  ERPROC: BIT     #BIT07,STATUS(R0) ;DONE BIT SET ?
006356  001402                          BEQ     ERPRC1          ;BR IF ORDER DIDN'T COMPLETE NORMALLY
006360  000137  007004                  JMP     DONE            ;PROCESS ERROR WITH 'DONE' BIT SET

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

006364  032760  010000  000016  ERPRC1: BIT     #BIT12,STATUS(R0) ;SEE IF DRIVE WAS UNSAFE
006372  001025                          BNE     PUNSAF          ;BR IF YES
006374  032760  004000  000016          BIT     #BIT11,STATUS(R0) ;PARITY ERROR OCCURRED
006402  001055                          BNE     UCPAR           ;BR IF IT DID
006404  032760  002000  000016          BIT     #BIT10,STATUS(R0) ;FATAL PARITY ERROR?
006412  001056                          BNE     FALPAR          ;BR IF THERE IS ONE
006414  032760  001000  000016          BIT     #BIT09,STATUS(R0) ;TIMEOUT?
006422  001076                          BNE     SWTIM           ;BR IF YES
006424  032760  040002  000016          BIT     #BIT14!BIT01,STATUS(R0)  ;DRIVE WENT OFFLINE ?
006432  001111                          BNE     OFLIN           ;BR IF IT DID
006434  032760  000004  000016          BIT     #BIT2,STATUS(R0)  ;PORT REQUEST TIME OUT ?
006442  001141                          BNE     PRTIM           ;BR IF IT DID
006444  000207                          RTS     PC              ;ERROR. RETURN

                                ;DRIVE IS PERSISTENTLY UNSAFE

006446  104401  001201          PUNSAF: TYPE    ,$CRLF          ;CR-LF
006452  104401  046222                  TYPE    ,EM12           ;'DRIVE UNSAFE' MESSAGE
006456  104401  053452                  TYPE    ,DRNUM          ;DRIVE NUMBER
006462  013746  001344                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
006466  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006470     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
006471     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
006472  104401  001201                  TYPE    ,$CRLF          ;CR-LF
006476  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
006502  104414  046222                  DISPLY  ,EM12           ;PERSISTENT DEVICE UNSAFE MESSAGE
006506  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
006512  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
006516  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF THE ERROR MESSAGE
006522  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
006526  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
006532  000137  026574                  JMP     DROP            ;DROP THE DRIVE

                                ;UNCORRECTABLE MASSBUS PARITY ERROR OCCURRED

006536  104401  001201          UCPAR:  TYPE    ,$CRLF          ;CR-LF
006542  104401  046124                  TYPE    ,EM10           ;'UNCORRECTABLE PARITY ERROR' MESSAGE
006546  000404                          BR      FALPR1          ;FINISH PROCESSING THE ERROR

                                ;'FATAL' MASSBUS PARITY ERROR OCCURRED

006550  104401  001201          FALPAR: TYPE    ,$CRLF          ;CR-LF
006554  104401  046167                  TYPE    ,EM11           ;'FATAL PARITY ERROR' MESSAGE
```

```
006560  104401  053452          FALPR1: TYPE    ,DRNUM          ;DRIVE NUMBER
006564  013746  001344                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
006570  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006572     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
006573     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
006574  104401  001201                  TYPE    ,$CRLF          ;CR-LF
006600  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
006604  032777  100000  172342          BIT     #SW15,@SWR      ;HALT ON ERROR ?
006612  001401                          BEQ     1$              ;BR IF NOT
006614  000000                          HALT                    ;ERROR HALT
006616  000207                  1$:     RTS     PC

                                ;SOFTWARE TIMEOUT OCCURRED

006620  004737  020176          SWTIM:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
006624  104414  046253                  DISPLY  ,EM13           ;PRINT THE TIME OUT MESSAGE
006630  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
006634  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
006640  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
006644  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
006650  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
006654  000207                          RTS     PC              ;RETURN

                                ;DRIVE WENT OFFLINE

006656  104401  001201          OFLIN:  TYPE    ,$CRLF          ;CR-LF
006662  104401  046325                  TYPE    ,EM14           ;'DRIVE WENT OFFLINE' MESSAGE
006666  104401  053452                  TYPE    ,DRNUM          ;DRIVE NUMBER
006672  013746  001344                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
006676  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006700     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
006701     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
006702  104401  001201                  TYPE    ,$CRLF          ;CR-LF
006706  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF THE ERROR MESSAGE
006712  104414  046325                  DISPLY  ,EM14           ;PRINT OFFLINE MESSAGE
006716  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF THE ERROR MESSAGE
006722  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF THE ERROR MESSAGE
006726  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF THE ERROR MESSAGE
006732  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
006736  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF THE ERROR MESSAGE
006742  000137  026574                  JMP     DROP            ;DROP THE DRIVE

                                ;PORT REQUEST TIMEOUT ERROR

006746  004737  020176          PRTIM:  JSR     PC,LINE1        ;TYPE LINE 1 OF THE ERROR MESSAGE
006752  104414  046350                  DISPLY  ,EM15           ;PRINT PORT TIME OUT MESSAGE
006756  004737  020242                  JSR     PC,LINE2        ;TYPE LINE 2 OF THE ERROR MESSAGE
006762  004737  020650                  JSR     PC,LINE3        ;TYPE LINE 3 OF THE ERROR MESSAGE
006766  004737  021320                  JSR     PC,LINE4        ;TYPE LINE 4 OF THE ERROR MESSAGE
006772  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
006776  004737  021730                  JSR     PC,LINE7        ;TYPE LINE 7 OF THE ERROR MESSAGE
007002  000207                          RTS     PC              ;RETURN

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

```
007004  032760  000030  000016  DONE:   BIT     #BIT04!BIT03,$STATUS(R0) ;UNSAFE OCCURRED
007012  001402                          BEQ     .+6                      ;BR IF NOT
007014  000137  012106                  JMP     UNSAF                    ;REPORT UNSAFE
007020  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)  ;IS 'WCE' SET ?
007026  001402                          BEQ     .+6                      ;BRANCH IF NOT SET
007030  000137  007766                  JMP     WCKER                    ;WRITE CHECK ERROR
007034  032760  040000  000246          BIT     #BIT14,$RMDS(R0) ;CHECK 'ERR'
007042  001002                          BNE     1$                       ;BR IF SET
007044  000137  011652                  JMP     TRFER                    ;PROCESS 'TRE
007050  032760  000400  000250  1$:     BIT     #BIT08,$RMER1(R0) ;'HCRC' SET?
007056  001402                          BEQ     .+6                      ;BR IF NOT
007060  000137  010314                  JMP     HCRCER                   ;PROCESS 'HCRC'
007064  032760  000020  000250          BIT     #BIT04,$RMER1(R0) ;'FMT' SET?
007072  001402                          BEQ     .+6                      ;BR IF NOT SET
007074  000137  010512                  JMP     CKFMT                    ;CHECK FORMAT ERROR
007100  032760  000200  000250          BIT     #BIT07,$RMER1(R0) ;'HCE' SET?
007106  001402                          BEQ     .+6                      ;BR IF NOT SET
007110  000137  010706                  JMP     CKHCE                    ;CHECK 'HCE' ERROR
007114  032760  020000  000250          BIT     #BIT13,$RMER1(R0) ;'OPI' SET?
007122  001402                          BEQ     .+6                      ;BR IF NOT SET
007124  000137  011206                  JMP     OPIER                    ;REPORT 'OPI'
007130  032760  000010  000250          BIT     #BIT3,$RMER1(R0) ;'PAR' SET?
007136  001402                          BEQ     .+6                      ;BR IF NOT SET
007140  000137  011340                  JMP     PARER                    ;REPORT 'PAR'
007144  032760  000040  000250          BIT     #BIT5,$RMER1(R0) ;'WCF' SET?
007152  001402                          BEQ     .+6                      ;BR IF NOT SET
007154  000137  012010                  JMP     WCFER                    ;REPORT 'WCF'
007160  032760  002000  000250          BIT     #BIT10,$RMER1(R0) ;'IAE' SET?
007166  001402                          BEQ     .+6                      ;BR IF NOT SET
007170  000137  011432                  JMP     IAEER                    ;REPORT 'IAE'
007174  032760  004000  000250          BIT     #BIT11,$RMER1(R0) ;'WLE' SET?
007202  001402                          BEQ     .+6                      ;BR IF NOT SET
007204  000137  011464                  JMP     WLEER                    ;REPORT 'WLE'
007210  032760  001000  000250          BIT     #BIT9,$RMER1(R0) ;'AOE' SET?
007216  001405                          BEQ     2$                       ;BR IF NOT SET
007220  032760  002000  000246          BIT     #BIT10,$RMDS(R0) ;'LST' SET?
007226  001401                          BEQ     2$                       ;BR IF NOT SET
007230  000207                          RTS     PC                       ;'AOE' & 'LST' SET, EXIT
007232  032760  010000  000250  2$:     BIT     #BIT12,$RMER1(R0)  ;SEE IF 'DTE' SET
007240  001402                          BEQ     .+6                      ;BR IF NOT
007242  000137  011316                  JMP     DTEER                    ;REPORT 'DTE' ERROR
007246  005760  000250                  TST     $RMER1(R0)               ;SEE IF 'DCK' SET
007252  100002                          BPL     .+6                      ;BR IF NOT
007254  000137  007312                  JMP     DCKER                    ;PROCESS 'DCK'
007260  032760  060000  000276          BIT     #BIT14!BIT13,$RMER2(R0) ;'SKI' OR 'OCYL' SET
007266  001006                          BNE     3$                       ;BRANCH IF SKI, OCYL SET
007270  032760  100000  000276          BIT     #BIT15,$RMER2(R0)  ;BAD SPOT ?
007276  001004                          BNE     4$                       ;BRANCH IF SO (NO, OTHER ERROR)
007300  000137  010460                  JMP     DRVER                    ;REPORT ERROR
007304  000137  011752          3$:     JMP     SKIER                    ;REPORT SKI ERROR
007310  000207                  4$:     RTS     PC                       ;EXIT FROM ERROR ANALYSIS  ROUT.

                                ;PROCESS DATA ('DCK') CHECK ERROR

007312  022760  010041  000300  DCKER:  CMP     #10041,$RMEC1(R0)  ;VALID POSITION COUNT ?
```

```
007320  101407                          BLOS    1$              ;BR IF NOT VALID
007322  005760  000300                  TST     $RMEC1(R0)      ;POSITION COUNT 0 ?
007326  001404                          BEQ     1$              ;BR IF 0'S
007330  005760  000302                  TST     $RMEC2(R0)      ;VALUE IN PATTERN REGISTER ?
007334  001027                          BNE     4$              ;BR IF YES
007336  000431                          BR      6$              ;DATA CHECK ERROR
007340  004737  020176          1$:     JSR     PC,LINE1        ;TYPE FIRST LINE OF ERROR MESSAGE
007344  104414  047755                  DISPLY  ,EM45           ;TYPE 'ECC LOGIC ERROR'
007350  004737  020242                  JSR     PC,LINE2        ;TYPE LINE 2 OF ERROR MESSAGE
007354  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007360  012737  000003  001350          MOV     #3,RETRY        ;RETRY COUNT
007366  004737  015134                  JSR     PC,$RETRY       ;RETRY THE ORDER
007372  000403                          BR      2$              ;RETRY WAS NOT SUCCESSFUL
007374  004737  021646                  JSR     PC,LINE6C       ;TYPE LINE 6C OF ERROR MESSAGE
007400  000402                          BR      3$              ;FINISH THE ERROR REPORT
007402  004737  021654          2$:     JSR     PC,LINE6D       ;TYPE LINE 6D OF ERROR MESSAGE
007406  004737  021730          3$:     JSR     PC,LINE7        ;TYPE LINE 7 OF ERROR MESSAGE
007412  000402                          BR      5$              ;EXIT
007414  004737  020060          4$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT A BAD SPOT ON THE PACK
007420  000207                  5$:     RTS     PC              ;IT IS, DON'T REPORT IT
007422  004737  020176          6$:     JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007426  104414  046425                  DISPLY  ,EM21           ;DATA CHECK ERROR
007432  004737  020242          DCKER1: JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007436  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
007442  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
007446  004737  014566                  JSR     PC,PRTBAD       ;SEE IF BAD SECTOR TO BE PRINTED
007452  012737  110100  001346          MOV     #BIT15!BIT12!BIT06,MASK  ;LOAD ERROR MASK
007460  032760  010100  000250          BIT     #BIT12!BIT06,$RMER1(R0)  ;CHECK 'DTE' & 'ECH'
007466  001003                          BNE     1$              ;BR IF SET
007470  004737  021620                  JSR     PC,LINE6        ;PRINT LINE 6 OF ERROR MESSAGE
007474  000477                          BR      8$              ;FINISH THE ERROR REPORT
007476  012737  000020  001350  1$:     MOV     #16.,RETRY      ;RETRY COUNT
007504  005001                          CLR     R1              ;R1 IS OFFSET CODE POINTER


007506  004737  016132          2$:     JSR     PC,GODRIV       ;RETRY
007512  005760  000016          3$:     TST     $STATUS(R0)     ;TEST FOR DONE
007516  001775                          BEQ     3$              ;BR IF NOT DONE
007520  100075                          BPL     10$             ;BR IF NOT ERROR
007522  032760  000200  000016          BIT     #BIT7,$STATUS(R0)  ;SEE IF ORDER TERMINIATED NORMALLY
007530  001006                          BNE     14$             ;BR IF NOT
007532  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007536  104414  052103                  DISPLY  ,LIN8M          ;'DIFFERENT ERROR DURING RETRY'
007542  000137  006364                  JMP     ERPRC1          ;SEE WHICH ERROR
007546  033760  001346  000250  14$:    BIT     MASK,$RMER1(R0) ;LOOK AT CURRENT ERROR
007554  001430                          BEQ     5$              ;BR IF DIFFERENT ERROR
007556  032760  010100  000250          BIT     #BIT12!BIT6,$RMER1(R0)  ;'ECH' OR 'DTE' STILL SET ?
007564  001437                          BEQ     7$              ;BR IF NEITHER SET
007566  105237  001351                  INCB    RETRY+1         ;INCREMENT RETRY COUNT
007572  123737  001350  001351          CMPB    RETRY,RETRY+1   ;DONE ?
007600  001342                          BNE     2$              ;BR IF NOT
007602  005201                          INC     R1              ;INCREMENT TABLE INDEX
007604  116137  002464  045533          MOVB    OFFCOD(R1),GENOPB+$FMT  ;OFFSET CODE
007612  001435                          BEQ     9$              ;BR IF END OF OFFSET TABLE
007614  062737  000002  001350          ADD     #2,RETRY        ;NEW RETRY LIMIT
007622  004737  015016                  JSR     PC,OFFST        ;OFFSET
```

```
007626  005737  045550          4$:     TST     GENOPB+$STATUS      ;SEE IF FINISHED WITH OFFSET
007632  001775                          BEQ     4$                  ;BR IF NOT
007634  100324                          BPL     2$                  ;BR IF NO ERROR PERFORMING OFFSET
007636  004737  022176          5$:     JSR     PC,LINE8            ;PRINT LINE 8 OF ERROR MESSAGE
007642  004737  023330          6$:     JSR     PC,INCHRD           ;INCREMENT 'HARD' ERROR COUNT
007646  004737  023424                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
007652  004737  021730                  JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
007656  004737  014566                  JSR     PC,PRTBAD           ;PRINT THE BAD SECTOR
007662  000436                          BR      13$                 ;CLEAN UP AND RETURN
007664  004737  021640          7$:     JSR     PC,LINE6B           ;PRINT LINE 6B OF ERROR MESSAGE
007670  004737  021556                  JSR     PC,LINE5B           ;PRINT LINE 5B OF THE ERROR MESSAGE
007674  004737  023304          8$:     JSR     PC,INCSOF           ;INCREMENT 'SOFT' ERROR COUNT
007700  004737  014026                  JSR     PC,ECC              ;CORRECT THE ERROR USING ECC AND CHECK IT
007704  000407                          BR      11$                 ;COMPARE THE BUFFER
007706  004737  021654          9$:     JSR     PC,LINE6D           ;PRINT LINE 6D OF ERROR MESSAGE
007712  000753                          BR      6$                  ;INCREMENT ERROR COUNT
007714  004737  021632          10$:    JSR     PC,LINE6A           ;PRINT LINE 6A OF ERROR MESSAGE
007720  004737  023304                  JSR     PC,INCSOF           ;INCREMENT 'SOFT' ERROR COUNT
007724  012737  000001  001376  11$:    MOV     #1,FRSTER           ;SET PROCESSING 'DCKER' INDICATOR
007732  004737  012634                  JSR     PC,CMPARD           ;COMPARE THE BUFFER
007736  105737  001377                  TSTB    FRSTER+1            ;ERROR IN COMPARE ?
007742  100406                          BMI     13$                 ;BRANCH IF ERROR
007744  004737  023424                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
007750  104414  052303                  DISPLY  ,LIN9G              ;'DATA COMPARE OK' MESSAGE
007754  004737  021730          12$:    JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
007760  004737  006136          13$:    JSR     PC,REFMT            ;REFORMAT THE ERROR SECTOR
007764  000207                          RTS     PC                  ;RETURN

                                ;WRITE CHECK ERROR PROCESSING

007766  032760  100000  000250  WCKER:  BIT     #BIT15,$RMER1(R0)   ;SEE IF 'DCK' SET ALSO
007774  001034                          BNE     2$                  ;BR IF IT IS
007776  004737  020176                  JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
010002  104414  046531                  DISPLY  ,EM23               ;PRINT WCE & DCK NOT
010006  005037  001346                  CLR     MASK                ;CLEAR ERROR MASK
010012  004737  020242                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
010016  004737  020650                  JSR     PC,LINE3            ;PRINT LINE 3 OF ERROR MESSAGE
010022  004737  021320                  JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
010026  004737  021410                  JSR     PC,LINE5            ;PRINT LINE 5 OF ERROR MESSAGE
010032  004737  023424                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
010036  012737  000003  001350          MOV     #3,RETRY            ;RETRY LIMIT
010044  004737  015134                  JSR     PC,$RETRY           ;RETRY THE OPERATION
010050  000403                          BR      1$                  ;RETRY UNSUCESSFUL
010052  004737  021646                  JSR     PC,LINE6C           ;PRINT LINE 6C OF ERROR MESSAGE
010056  000502                          BR      8$                  ;FINISH PROCESSING THE ERROR
010060  004737  021654          1$:     JSR     PC,LINE6D           ;PRINT LINE 6D OF ERROR MESSAGE
010064  000506                          BR      10$                 ;FINISH PROCESSING THE ERROR
010066  004737  020060          2$:     JSR     PC,SPOTCK           ;SEE IF ERROR AT BAD SPOT ON THE PACK
010072  000507                          BR      11$                 ;EXIT IF AT BAD SPOT ON PACK
010074  004737  020176                  JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
010100  012737  046456  010126          MOV     #EM22,13$           ;ASSUME THAT EM22 WILL BE PRINTED
010106  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)   ;DID 'WCK' ALSO SET ?
010114  001003                          BNE     12$                 ;BR IF IT DID
010116  012737  047357  010126          MOV     #EM37,13$           ;MESSAGE FOR 'DCK' AND 'WCK' NOT DURING
                                                                    ;WRITE CHECK
010124  104414                  12$:    DISPLY                      ;TYPE THE ERROR MESSAGE
```

```
010126  000000                  13$:    .WORD   0                       ;MESSAGE ADDRESS GOES HERE
010130  004737  020242                  JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
010134  004737  020650                  JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
010140  004737  021320                  JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
010144  004737  021410                  JSR     PC,LINE5                ;PRINT LINE 5 OF ERROR MESSAGE
010150  032760  000100  000250          BIT     #BIT06,$RMER1(R0)       ;ECH SET ALSO ?
010156  001442                          BEQ     8$                      ;FINISH PROCESSING THE ERROR
010160  012737  000020  001350  3$:     MOV     #16.,RETRY              ;RETRY LIMIT - 16 (10)
010166  004737  016132          4$:     JSR     PC,GODRIV               ;RETRY THE ORDER
010172  005760  000016          5$:     TST     STATUS(R0)              ;ORDER FINISHED ?
010176  001775                          BEQ     5$                      ;BR IF NOT
010200  100405                          BMI     6$                      ;BR IF ERROR ON ORDER
010202  105237  001351                  INCB    RETRY+1                 ;INCREMENT RETRY COUNT
010206  004737  021646                  JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
010212  000431                          BR      9$                      ;FINISH ERROR PROCESSING
010214  105237  001351          6$:     INCB    RETRY+1                 ;INCREMENT RETRY COUNT
010220  123737  001350  001351          CMPB    RETRY,RETRY+1           ;DONE ?
010226  001714                          BEQ     1$                      ;BR IF AT RETRY LIMIT
010230  032760  100000  000250          BIT     #BIT15,$RMER1(R0)       ;'DCK' SET
010236  001407                          BEQ     7$                      ;BR IF NOT - DIFFERENT ERROR
010240  032760  000100  000250          BIT     #BIT06,$RMER1(R0)       ;'ECH' ALSO SET ?
010246  001347                          BNE     4$                      ;BR IF IT IS, RETRY ORDER
010250  004737  021646                  JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
010254  000403                          BR      8$                      ;FINISH PROCESSING ERROR
010256  004737  022176          7$:     JSR     PC,LINE8                ;PRINT LINE 8 - 'DIFFERENT ERROR '
010262  000405                          BR      9$                      ;FINISH PROCESSING ERROR
010264  004737  023424          8$:     JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
010270  004737  021730                  JSR     PC,LINE7                ;FINISH THE ERROR MESSAGE
010274  000406                          BR      11$                     ;EXIT
010276  004737  023424          9$:     JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
010302  004737  021730          10$:    JSR     PC,LINE7                ;FINISH THE ERROR MESSAGE
010306  004737  006136                  JSR     PC,REFMT                ;REFORMAT THE SECTOR IN ERROR
010312  000207                  11$:    RTS     PC                      ;RETURN

                                ;REPORT 'HCRC' ERROR

010314  004737  020060          HCRCER: JSR     PC,SPOTCK               ;SEE IF ERROR AT PACK BAD SPOT
010320  000456                          BR      3$                      ;EXIT IF IT IS
010322  004737  022262                  JSR     PC,READDR               ;READ ERROR SECTOR HEADER
010326  004737  015042                  JSR     PC,READHD               ;GET THE HEAD INFORMATION
010332  004737  020176                  JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
010336  104414  046404                  DISPLY  ,EM20                   ;REPORT 'HCRC'
010342  004737  020242                  JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
010346  004737  020650                  JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
010352  004737  021320                  JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
010356  004737  021510                  JSR     PC,LINE5A               ;PRINT THE HEADER INFORMATION
010362  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)       ;'WCE' ERROR ALSO ?
010370  001402                          BEQ     1$                      ;BR IF NOT
010372  004737  021410                  JSR     PC,LINE5                ;DISPLAY WORDS WHICH CAUSED 'WCE'
010376  004737  023304          1$:     JSR     PC,INCSOF               ;INCREMENT 'SOFT' ERROR COUNT
010402  004737  023424                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
010406  012737  000400  001346          MOV     #BIT8,MASK              ;SET ERROR MASK
010414  012737  000003  001350          MOV     #3,RETRY                ;RETRY LIMIT
010422  004737  015134                  JSR     PC,$RETRY               ;RETRY ORDER
010426  000405                          BR      2$                      ;RETRY NOT SUCESSFUL
010430  004737  021646                  JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
```

```
010434  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
010440  000176                          BR      3$              ;EXIT
010442  004737  021654          2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
010446  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
010452  004737  006136                  JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
010456  000207                  3$:     RTS     PC              ;RETURN

                                ;REPORT DRIVE ERROR

010460  004737  020176          DRVER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
010464  104414  047021                  DISPLY  ,EM30           ;REPORT DRIVE ERROR
010470  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
010474  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
010500  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
010504  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
010510  000207                          RTS     PC              ;RETURN

                                ;PROCESS FORMAT ('FER') ERROR

010512  032760  000400  000250  CKFMT:  BIT     #BIT8,$RMER1(R0) ;'HCRC' SET ON ORIGINAL ERROR ?
010520  001402                          BEQ     1$              ;BR IF NOT SET
010522  000137  010314                  JMP     HCRCER          ;REPORT HCRC ERROR
010526  004737  022262          1$:     JSR     PC,READOR       ;GET CORRECTED TRACK & SECTOR ADDRSSES
010532  004737  015042                  JSR     PC,READHD       ;READ HEADER
010536  032737  000400  045566          BIT     #BIT8,GENREG+RMER1 ;'HCRC' SET WHEN HEADER READ
010544  001002                          BNE     2$              ;BR IF 'HCRC' SET
010546  000137  011512                  JMP     FMTER           ;NO, ERROR IS 'FMT' ONLY
010552  004737  020060          2$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT ON THE PACK
010556  000452                          BR      5$              ;EXIT IF IT IS
010560  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
010564  104414  046610                  DISPLY  ,EM24           ;HEADER READ ERROR - FMT BIT DROPPED UP
010570  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
010574  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
010600  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
010604  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
010612  001402                          BEQ     3$              ;BR IF NOT
010614  004737  021410                  JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
010620  004737  021510          3$:     JSR     PC,LINE5A       ;DISPLAY HEADER
010624  004737  023304                  JSR     PC,INCSOF       ;INCREMENT SOFT ERROR COUNT
010630  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
010634  012737  000020  001346          MOV     #BIT4,MASK      ;SET ERROR MASK
010642  012737  000003  001350          MOV     #3,RETRY        ;RETRY LIMIT
010650  004737  015134                  JSR     PC,$RETRY       ;RETRY THE ORDER
010654  000405                          BR      4$              ;RETRY NOT SUCESSFUL
010656  004737  021646                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
010662  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
010666  000406                          BR      5$              ;EXIT
010670  004737  021654          4$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
010674  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
010700  004737  006136                  JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
010704  000207                  5$:     RTS     PC              ;RETURN

                                ;PROCESS HEADER COMPARE ('HCE') ERROR

010706  032760  000400  000250  CKHCE:  BIT     #BIT8,$RMER1(R0) ;HCRC SET ON ORIGINAL ERROR ?
010714  001402                          BEQ     1$              ;BR IF NOT SET
```

```
010716  000137  010314                  JMP     HCRCER          ;REPORT HEADER CRC ERROR
010722  004737  022262          1$:     JSR     PC,READDR       ;GET CURRENT SECTOR & TRACK ADDRS
010726  004737  015042                  JSR     PC,READHD       ;READ HEADER OF CURRENT SECTOR
010732  032737  000400  045566          BIT     #BIT8,GENREG+RMER1 ;'HCRC' SET ?
010740  001016                          BNE     3$              ;BR IF SET
010742  042737  150000  056150          BIC     #150000,CYLDER  ;CLEAR FORMAT,MFG,USER BITS FROM HEADER
010750  026037  000270  056150          CMP     $RMDC(R0),CYLDER  ;CORRECT CYLINDER ?
010756  001402                          BEQ     2$              ;BR IF IT IS
010760  000137  011132                  JMP     POSER           ;REPORT POSITIONING ERROR
010764  052737  150000  056150  2$:     BIS     #150000,CYLDER  ;RESTORE THE FORMAT,MFG,USER BITS
010772  000137  011570                  JMP     HCEER           ;REPORT 'HCE' ERROR
010776  004737  020060          3$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
011002  000452                          BR      6$              ;EXIT IF IT IS
011004  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011010  104414  046656                  DISPLY  ,EM25           ;HEADER READ ERROR - 'HCE' SET
011014  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011020  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011024  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011030  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
011036  001402                          BEQ     4$              ;BR IF NOT
011040  004737  021410                  JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
011044  004737  021510          4$:     JSR     PC,LINE5A       ;PRINT LINE 5 OF ERROR MESSAGE
011050  004737  023304                  JSR     PC,INCSOF       ;INCREMENT SOFT ERROR COUNT
011054  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011060  012737  000200  001346          MOV     #BIT7,MASK      ;SET ERROR MASK
011066  012737  000003  001350          MOV     #3,RETRY        ;RETRY LIMIT
011074  004737  015134                  JSR     PC,$RETRY       ;RETRY THE ORDER
011100  000405                          BR      5$              ;RETRY NOT SUCESSFUL
011102  004737  021646                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
011106  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011112  000406                          BR      6$              ;EXIT
011114  004737  021654          5$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
011120  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011124  004737  006136                  JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
011130  000207                  6$:     RTS     PC              ;RETURN

                                ;REPORT POSSIBLE POSITIONING ERROR

011132  004737  014764          POSER:  JSR     PC,RECALT       ;RECALIBRATE
011136  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011142  104414  050153                  DISPLY  ,EM51           ;PROGRAM DETECTED POSITIONING ERROR
011146  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011152  004737  020676                  JSR     PC,LINE3C       ;PRINT LINE 3C OF ERROR MESSAGE
011156  052737  150000  056150          BIS     #150000,CYLDER  ;RESTORE THE FORMAT BIT
011164  004737  021510                  JSR     PC,LINE5A       ;PRINT LINE 5A OF THE ERROR MESSAGE
011170  004737  023400                  JSR     PC,INCMIS       ;INCREMENT MISPOSITIONING COUNT
011174  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011200  004737  022056                  JSR     PC,LINE7A       ;PRINT LINE 7A OF ERROR MESSAGE
011204  000207                          RTS     PC              ;EXIT

                                ;REPORT 'OPI' ERROR

011206  004737  020060          OPIER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
011212  000207                          RTS     PC              ;RETURN IF IT IS
011214  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011220  104414  047053                  DISPLY  ,EM31           ;'OPI' ERROR
```

```
011224  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011230  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011234  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011240  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011244  012737  020000  001346          MOV     #BIT13,MASK     ;ERROR MASK
011252  012737  000003  001350  OPIER1: MOV     #3,RETRY        ;RETRY LIMIT
011260  004737  015134                  JSR     PC,$RETRY       ;RETRY THE ORDER
011264  000405                          BR      1$              ;RETRY UNSUCESSFUL
011266  004737  021646                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
011272  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011276  000207                          RTS     PC              ;EXIT
011300  004737  021654          1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
011304  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011310  004737  006136                  JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
011314  000207                          RTS     PC              ;RETURN

                                ;REPORT 'DTE' ERROR

011316  004737  020060          DTEER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
011322  000207                          RTS     PC              ;RETURN IF IT IS
011324  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011330  104414  047116                  DISPLY  ,EM32           ;'DTE' ERROR
011334  000137  007432                  JMP     DCKER1          ;FINISH PROCESSING THE 'DTE' ERROR

                                ;REPORT 'PAR' ERROR

011340  004737  020176          PARER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011344  104414  047151                  DISPLY  ,EM33           ;REPORT 'PAR'
011350  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011354  004737  020754                  JSR     PC,LINE3E       ;PRINT LINE 3E OF ERROR MESSAGE
011360  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011364  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011370  012737  000010  001346          MOV     #BIT03,MASK     ;ERROR MASK
011376  012737  000003  001350          MOV     #3,RETRY        ;RETRY LIMIT
011404  004737  015134                  JSR     PC,$RETRY       ;RETRY ORDER
011410  000405                          BR      2$              ;RETRY UNSUCESSFUL
011412  004737  021646                  JSR     PC,LINE6C       ;RETRY SUCESSFUL
011416  004737  021730          1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011422  000207                          RTS     PC              ;EXIT
011424  004737  021654          2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
011430  000772                          BR      1$              ;FINISH ERROR MESSAGE

                                ;REPORT 'IAE' ERROR

011432  004737  020176          IAEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011436  104414  047270                  DISPLY  ,EM35           ;REPORT 'IAE'
011442  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011446  004737  021042                  JSR     PC,LINE3F       ;PRINT LINE 3F OF ERROR MESSAGE
011452  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011456  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011462  000207                          RTS     PC              ;RETURN

                                ;REPORT 'WLE' ERROR

011464  004737  020176          WLEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011470  104414  047326                  DISPLY  ,EM36           ;REPORT 'WLE'
```

```
011474  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011500  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011504  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011510  000207                          RTS     PC              ;RETURN

                                ;REPORT FORMAT ERROR

011512  004737  020176          FMTER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011516  104414  046737                  DISPLY  ,EM26           ;FORMAT ERROR
011522  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011526  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011532  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011536  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
011544  001402                          BEQ     1$              ;BR IF NOT
011546  004737  021410                  JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
011552  004737  021510          1$:     JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
011556  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011562  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011566  000207                          RTS     PC

                                ;REPORT HEADER COMPARE ERROR

011570  004737  020176          HCEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011574  104414  046764                  DISPLY  ,EM27           ;HEADER COMPARE ERROR
011600  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011604  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011610  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011614  032760  040000  000244          BIT     #BIT14,$RMCS2(R0)  ;'WCE' ERROR ALSO ?
011622  001402                          BEQ     1$              ;BR IF NOT
011624  004737  021410                  JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
011630  004737  021510          1$:     JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
011634  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011640  004737  021730                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011644  004737  006136                  JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
011650  000207                          RTS     PC              ;RETURN

                                ;PROCESS CONTROL/INTERFACE TRANSFER ERROR

011652  004737  020176          TRFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011656  104414  047441                  DISPLY  ,EM40           ;RH11 OR UNIBUS TRANSFER ERROR
011662  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011666  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011672  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011676  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011702  032760  121400  000244          BIT     #BIT15!BIT13!BIT9!BIT8,$RMCS2(R0)  ;'DLT','UPE','MXF', 'MDPE' SET ?
011710  001415                          BEQ     2$              ;BR IF NONE SET
011712  012737  000003  001350          MOV     #3,RETRY        ;RETRY LIMIT
011720  005037  001346                  CLR     MASK            ;CLEAR ERROR MASK
011724  004737  015134                  JSR     PC,$RETRY       ;RETRY THE OPERATION
011730  000403                          BR      1$              ;RETURN HERE IF RETRY UNSUCESSFUL
011732  004737  021646                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
011736  000402                          BR      2$              ;FINISH THE ERROR REPORT
011740  004737  021654          1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
011744  004737  021730          2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
011750  000207                          RTS     PC
```

```
                                        ;PROCESS 'SKI' OR 'OCYL' ERRORS

011752  004737  020176          SKIER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011756  104414  050115                  DISPLY  ,EM50           ;'SKI' OR 'OCYL' ERROR
011762  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011766  004737  020664                  JSR     PC,LINE3B       ;PRINT LINE 3B OF ERROR MESSAGE
011772  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011776  004737  023354                  JSR     PC,INCSKI       ;INCREMENT 'SKI' OR 'OCYL' ERROR COUNT
012002  004737  022056                  JSR     PC,LINE7A       ;PRINT LINE 7A OF ERROR MESSAGE
012006  000207                          RTS     PC

                                        ;REPORT WRITE CLOCK FAILURE ('WCF')

012010  004737  020176          WCFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012014  104414  047226                  DISPLY  ,EM34           ;REPORT WRITE CLOCK FAILURE
012020  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012024  004737  020656                  JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
012030  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012034  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012040  004737  014566                  JSR     PC,PRTBAD       ;SEE IF BAD SECTOR TO BE PRINTED
012044  012737  000003  001350          MOV     #3,RETRY        ;RETRY COUNT
012052  012737  000040  001346          MOV     #BITO5,MASK     ;ERROR MASK
012060  004737  015134                  JSR     PC,$RETRY       ;RETRY THE ORDER
012064  000405                          BR      2$              ;RETURN HERE IF RETRY UNSUCESSFUL
012066  004737  021646                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
012072  004737  021730          1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012076  000207                          RTS     PC
012100  004737  021654          2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012104  000772                          BR      1$

                                        ;PROCESS DRIVE UNSAFE ERROR

012106  004737  020176          UNSAF:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012112  104414  050216                  DISPLY  ,EM60           ;REPORT DRIVE UNSAFE
012116  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012122  004737  020650                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012126  004737  023424                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012132  012737  000003  001350          MOV     #3,RETRY        ;RETRY COUNT
012140  004737  015134                  JSR     PC,$RETRY       ;RETRY THE ORDER
012144  000403                          BR      1$              ;RETRY WAS UNSUCESSFUL
012146  004737  021646                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
012152  000402                          BR      2$              ;CONTINUE WITH ERROR REPORT
012154  004737  021654          1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012160  004737  021730          2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012164  000207                          RTS     PC              ;RETURN

                                        ;REPORT AN 'UNKNOWN' DATA PATTERN

012166  105737  001376          NOMTCH: TSTB    FRSTER          ;FIRST ERROR IN THE SECTOR ?
012172  001013                          BNE     1$              ;BR IF NOT OR IF PROCESSING 'DCKER'
012174  004737  020176                  JSR     PC,LINE1        ;TYPE LINE 1 OF ERROR MESSAGE
012200  104414  047624                  DISPLY  ,EM43           ;'CAN'T MATCH DATA WITH PATTERN'
012204  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012210  004737  020656                  JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
012214  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012220  000404                          BR      2$              ;CONTINUE PROCESSING ERROR
```

```
012222  104414  047624          1$:     DISPLY  ,EM43          ;'CAN'T MATCH DATA WITH PATTERN'
012226  104414  001201                  DISPLY  ,$CRLF         ;CR-LF
012232  104414  052233          2$:     DISPLY  ,LIN9I         ;HEADER FOR DATA PRINTOUT
012236  010146                          MOV     R1,-(SP)       ;ADDRESS OF WORD 1
012240  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 1
012244  104414  052716                  DISPLY  ,LINSP         ;SPACES
012250  012146                          MOV     (R1)+,-(SP)    ;ADDRESS OF WORD 1
012252  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 1
012256  104414  001201                  DISPLY  ,$CRLF         ;CR-LF
012262  010146                          MOV     R1,-(SP)       ;ADDRESS OF WORD 2
012264  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 2
012270  104414  052716                  DISPLY  ,LINSP         ;SPACES
012274  012146                          MOV     (R1)+,-(SP)    ;ADDRESS OF WORD 2
012276  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 2
012302  104414  001201                  DISPLY  ,$CRLF         ;CR-LF
012306  010146                          MOV     R1,-(SP)       ;ADDRESS OF WORD 3
012310  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 3
012314  104414  052716                  DISPLY  ,LINSP         ;SPACES
012320  012146                          MOV     (R1)+,-(SP)    ;ADDRESS OF WORD 3
012322  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 3
012326  104414  001201                  DISPLY  ,$CRLF         ;CR-LF
012332  010146                          MOV     R1,-(SP)       ;ADDRESS OF WORD 4
012334  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 4
012340  104414  052716                  DISPLY  ,LINSP         ;SPACES
012344  012146                          MOV     (R1)+,-(SP)    ;ADDRESS OF WORD 4
012346  004737  022210                  JSR     PC,LINOCT      ;TYPE WORD 4
012352  104414  001201                  DISPLY  ,$CRLF         ;CR-LF
012356  062701  000770                  ADD     #<252.*2.>,R1  ;INCREMENT BUFFER POINTER
012362  162737  000400  001410          SUB     #256.,CMCNT    ;ADJUST WORD COUNT FOR MATCH
012370  132760  000001  000024          BITB    #BITOO,$CODE(R0) ;HEADER OPERATION INVOLVED ?
012376  001405                          BEQ     3$             ;NO
012400  062701  000004                  ADD     #4,R1          ;ADJUST BUFFER ADDRESS
012404  162737  000002  001410          SUB     #2,CMCNT       ;ADJUST WORD COUNT
012412  005002                  3$:     CLR     R2             ;CLEAR 'WORDS TO COMPARE' COUNT IN R2
012414  112737  000001  001376          MOVB    #1,FRSTER      ;SET 'NOT FIRST ERROR' INDICATOR
012422  112737  177777  001377          MOVB    #-1,FRSTER+1   ;SET ERROR FOUND INDICATOR
012430  013737  001472  001406          MOV     CMPLMT,LIMIT   ;RESET THE COMPARE ERROR TYPEOUT LIMIT
012436  000207                          RTS     PC             ;RETURN

                                ;CHECK ERROR BITS IN THE RH11 & RM03 REGISTERS

012440  032760  060000  000234  CKERR:  BIT     #60000,$RMCS1(R0)  ;SEE IF 'TRE' OR 'MCPE' SET
012446  001012                          BNE     1$             ;BR IT EITHER SET
012450  032760  177400  000244          BIT     #177400,$RMCS2(R0)  ;SEE IF ERROR BITS IN CS2 SET
012456  001006                          BNE     1$             ;BR IF ANY SET
012460  005760  000250                  TST     $RMER1(R0)     ;ANY BITS SET IN ER1
012464  001003                          BNE     1$             ;BR IF ANY SET
012466  005760  000276                  TST     $RMER2(R0)     ;ANY BITS SET IN ER2 ?
012472  001416                          BEQ     2$             ;BR IF NONE SET
012474  004737  020176          1$:     JSR     PC,LINE1       ;PRINT LINE 1 OF ERROR MESSAGE
012500  104414  047671                  DISPLY  ,EM44          ;ERROR BITS SET, BUT 'SC' OR 'TRE' NOT SET
012504  004737  020242                  JSR     PC,LINE2       ;PRINT LINE 2 OF ERROR MESSAGE
012510  004737  020650                  JSR     PC,LINE3       ;PRINT LINE 3 OF ERROR MESSAGE
012514  004737  021320                  JSR     PC,LINE4       ;PRINT LINE 4 OF ERROR MESSAGE
012520  004737  023424                  JSR     PC,INCTOT      ;INCREMENT TOTAL ERROR COUNT
012524  004737  021730                  JSR     PC,LINE7       ;PRINT LINE 7 OF ERROR MESSAGE
```

```
012530  000207                  2$:     RTS     PC                  ;RETURN

                                ;CHECK BUS ADDRESS REGISTER & WORD COUNT REGISTER

012532  005760  000236          CKBUS:  TST     $RMWC(R0)           ;CHECK WORD COUNT
012536  001010                          BNE     1$                  ;BR IF NOT ZERO
012540  016046  000020                  MOV     $WRDL(R0),-(SP)     ;WORD LENGTH
012544  006316                          ASL     (SP)                ;CHANGE INTO BYTE COUNT
012546  066016  000006                  ADD     $BUF(R0),(SP)       ;ADD THE STARTING LOCATION
012552  022660  000240                  CMP     (SP)+,$RMBA(R0)     ;BUFFER ADDRESS PROPER ?
012556  001416                          BEQ     2$                  ;BR IF OK
012560  004737  020176          1$:     JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
012564  104414  047477                  DISPLY  ,EM41               ;BUS ADDRESS OR WORD COUNT INCORRECT
012570  004737  020242                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
012574  004737  020706                  JSR     PC,LINE3D           ;PRINT LINE 3D OF ERROR MESSAGE
012600  004737  021320                  JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
012604  004737  023424                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
012610  004737  021730                  JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
012614  000207                  2$:     RTS     PC

                                ;COMPARE THE BUFFER

012616  132760  000004  000024  CMPAR:  BITB    #BIT02,$CODE(R0)    ;SEE IF READ ORDER
012624  001001                          BNE     1$                  ;BR IF IT IS
012626  000207                          RTS     PC                  ;RETURN
012630  005037  001376          1$:     CLR     FRSTER              ;CLEAR 'FIRST ERROR' INDICATOR
012634  032777  000002  166312  CMPARD: BIT     #SW01,@SWR          ;IS SWITCH 1 SET?
012642  001401                          BEQ     1$                  ;BR IF NOT
012644  000207                          RTS     PC                  ;YES, DON'T COMPARE
012646  005037  001404          1$:     CLR     ERCTR               ;CLEAR THE ERROR COUNTER
012652  016001  000006                  MOV     $BUF(R0),R1         ;BUFFER ADDRESS
012656  016037  000020  001410          MOV     $WRDL(R0),CMCNT     ;WORD COUNT TO WORKING LOCATION
012664  066037  000236  001410          ADD     $RMWC(R0),CMCNT     ;CALCULATE ACTUAL WORDS TRANSFERED
012672  016037  000012  001412          MOV     $CYL(R0),CMCYL      ;CYLINDER ADDRESS WORKING LOCATION
012700  052737  010000  001412          BIS     #BIT12,CMCYL        ;SET FORMAT BIT
012706  052737  140000  001412          BIS     #140000,CMCYL       ;SET MFG AND USER BITS
012714  016037  000010  001414          MOV     $SEC(R0),CMSEC      ;SECTOR & TRACK ADDRESSES TO WORKING LOCNS
012722  013737  001472  001406          MOV     CMPLMT,LIMIT        ;DISPLAY LIMIT
012730  005237  001406                  INC     LIMIT               ;CONVERT PARAMETER INTO LIMIT VALUE
012734  012737  177777  001374  CMSTR:  MOV     #-1,ZROIND          ;CLEAR THE 'ZERO'S' INDICATOR
012742  005037  001400                  CLR     SAVER1              ;CLEAR THE R1 SAVE WORD
012746  005037  001402                  CLR     SAVER5              ;CLEAR THE R5 SAVE WORD
012752  023760  001410  000022          CMP     CMCNT,$SSEC(R0)     ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
012760  101005                          BHI     1$                  ;BR IF IT IS
012762  013702  001410                  MOV     CMCNT,R2            ;LESS THAN, USE REMAINING BUFFER
012766  005037  001410                  CLR     CMCNT               ;SET COUNTER TO 0
012772  000405                          BR      2$                  ;
012774  016002  000022          1$:     MOV     $SSEC(R0),R2        ;COMPARE SECTOR
013000  166037  000022  001410          SUB     $SSEC(R0),CMCNT     ;DECREMENT WORD COUNT
013006  126027  000024  000005  2$:     CMPB    $CODE(R0),#5        ;READ HEADER & DATA?
013014  001023                          BNE     CMDAT               ;BR IF NOT
013016  023721  001412          CMHED:  CMP     CMCYL,(R1)+         ;CHECK CYLINDER
013022  001402                          BEQ     1$                  ;BR IF COMPARE OK
013024  004737  013052                  JSR     PC,CMSTR2                   ;REPORT ERROR
013030  023721  001414          1$:     CMP     CMSEC,(R1)+         ;COMPARE SECTOR & TRACK
013034  001402                          BEQ     4$                  ;BR IF EQ
```

```
013036  004737  013052                  JSR     PC,CMSTR2               ;REPORT ERROR
013042  162702  000002          4$:     SUB     #2,R2           ;SUBTRACT HEADER LENGTH FROM SIZE
013046  003564                          BLE     CMPRX           ;BR IF FINISHED
013050  000405                          BR      CMDAT           ;COMPARE THE DATA PORTION
013052  005237  001404          CMSTR2: INC     ERCTR           ;INCREMENT THE ERROR COUNT
013056  004737  013426                  JSR     PC,CMPRT        ;REPORT THE COMPARISON ERROR
013062  000207                          RTS     PC              ;CHECK THE REST OF THE HEADER
013064  004737  013750          CMDAT:  JSR     PC,MATCH        ;FIND THE PATTERN
013070  000403                          BR      2$              ;FOUND A PATTERN
013072  004737  012166                  JSR     PC,NOMTCH       ;RETURN HERE IF NO MATCH WITH PATTERN MADE
013076  000456                          BR      8$              ;BYPASS COMPARE ROUTINE
013100  011405                  2$:     MOV     (R4),R5         ;ADDRESS OF PATTERN ADDRESS IN R4
013102  012703  000020                  MOV     #20,R3          ;R3 IS PATTERN POS COUNTER
013106  022125                  3$:     CMP     (R1)+,(R5)+     ;COMPARE BUFFER WITH PATTERN
013110  001016                          BNE     5$              ;BR IF NOT EQUAL
013112  005737  001404                  TST     ERCTR           ;ERRORS DETECTED ?
013116  001406                          BEQ     4$              ;BR IF NO ERRORS
013120  032777  000010  166026          BIT     #SW3,@SWR       ;SWITCH 3 SET ?
013126  001402                          BEQ     4$              ;BR IF NOT SET
013130  004737  013426                  JSR     PC,CMPRT        ;DISPLAY THE WORD
013134  005302                  4$:     DEC     R2              ;DECREMENT SIZE COUNT
013136  003436                          BLE     8$              ;BR WHEN AT END
013140  005303                          DEC     R3              ;DECREMENT PATT POS COUNT
013142  001361                          BNE     3$              ;BR IF NOT AT END OF PATT
013144  000755                          BR      2$              ;RESTART THE PATTERN
013146  005761  177776          5$:     TST     -2(R1)          ;IS MISCOMPARED CHARACTER=0
013152  001410                          BEQ     6$              ;BR IF YES
013154  012737  177777  001374          MOV     #-1,ZROIND      ;SET NON-ZERO MISCOMPARED INDCATOR
013162  005237  001404                  INC     ERCTR           ;INCREMENT THE ERROR COUNTER
013166  004737  013426                  JSR     PC,CMPRT        ;REPORT ERROR
013172  000760                          BR      4$              ;CONTINUE COMPARE
013174  105737  001376          6$:     TSTB    FRSTER          ;FIRST ERROR?
013200  100407                          BMI     7$              ;BR IF NOT
013202  005037  001374                  CLR     ZROIND          ;SET THE ZERO INDICATOR
013206  010137  001400                  MOV     R1,SAVER1       ;SAVE CURRENT R1
013212  010537  001402                  MOV     R5,SAVER5       ;SAVE CURRENT R5
013216  000746                          BR      4$              ;CONTINUE COMPARE
013220  005737  001374          7$:     TST     ZROIND          ;ANY MISCOMPARIONS NOT ZEROS ?
013224  001743                          BEQ     4$              ;BR IF NONE-ALL ERRORS=ZERO
013226  004737  013426                  JSR     PC,CMPRT        ;REPORT ERROR
013232  000740                          BR      4$              ;CONTINUE COMPARING
013234  023727  001410  000004  8$:     CMP     CMCNT,#4        ;LAST 3WORDS ?
013242  002466                          BLT     CMPRX           ;YES
013244  126027  000024  000005          CMPB    $CODE(R0),#5    ;READ HEAD AND DATA ?
013252  001414                          BEQ     9$              ;YES
013254  013702  001410          13$:    MOV     CMCNT,R2        ;SET COUNTER = REMAIN BUFFER LENGTH
013260  020227  000004                  CMP     R2,#4           ;LAST  3 WORDS ?
013264  002455                          BLT     CMPRX           ;YES,EXIT
013266  162737  000400  001410          SUB     #256.,CMCNT     ;GREATER THAN A SECTOR ?
013274  003673                          BLE     CMDAT           ;NO,RETURN TO COMPARE LOOP
013276  012702  000400                  MOV     #256.,R2        ;SET COUNTER =SECTOR SIZE
013302  000670                          BR      CMDAT           ;RETURN TO COMPARE LOOP
013304  105237  001414          9$:     INCB    CMSEC           ;INCREMENT COUNTER
013310  123727  001414  000040          CMPB    CMSEC,#32.      ;MAX SECTOR # ?
013316  103421                          BLO     10$             ;NO
013320  105037  001414                  CLRB    CMSEC           ;RESET SECTOR #
```

```
013324  105237  001415                  INCB    CMTRK           ;INCREMENT TRACK #
013330  123727  001415  000005          CMPB    CMTRK,#5        ;MAX TRACK # ?
013336  103411                          BLO     10$             ;NO
013340  105037  001415                  CLRB    CMTRK           ;RESET TRACK #
013344  005237  001412                  INC     CMCYL           ;INCREMENT CYLINDER NUMBER
013350  023727  001412  151466          CMP     CMCYL,#150000+822.      ;LAST CYLINDER ?
013356  103401                          BLO     10$             ;NO
013360  000417                          BR      CMPRX           ;NORMAL RETURN,NOT WRAP AROUND
013362  023721  001412          10$:    CMP     CMCYL,(R1)+     ;COMPARE 1ST HEADER WORD
013366  001402                          BEQ     11$             ;MATCH
013370  004737  013052                  JSR     PC,CMSTR2       ;NOT MATCH
013374  023721  001414          11$:    CMP     CMSEC,(R1)+     ;SECOND WORD OF HEADER
013400  001402                          BEQ     12$             ;MATCH
013402  004737  013052                  JSR     PC,CMSTR2       ;NOT MATCH
013406  162737  000002  001410  12$:    SUB     #2,CMCNT                ;ADJUST WORD COUNT
013414  003401                          BLE     CMPRX           ;COMPARE IS DONE
013416  000716                          BR      13$             ;RETURN TO COMPARE LOOP

013420  004737  013702          CMPRX:  JSR     PC,ENDCMP       ;PRINT LAST LINE IF ERRORS
013424  000207                          RTS     PC

                                ;TYPE DATA COMPARE ERRORS

013426  005737  001400          CMPRT:  TST     SAVER1          ;PRINT SAVED VALUES ?
013432  001010                          BNE     2$              ;BR IF NOT
013434  105737  001376                  TSTB    FRSTER          ;FIRST ERROR?
013440  100402                          BMI     1$              ;BR IF NOT
013442  004737  013522                  JSR     PC,4$           ;PRINT INITIAL MESSAGE INFO
013446  004737  013604          1$:     JSR     PC,8$           ;PRINT REMAINDER OF MESSAGE
013452  000422                          BR      3$              ;EXIT
013454                          2$:
013454  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013456  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
013460  013701  001400                  MOV     SAVER1,R1       ;DISPLAY SAVED R1
013464  013705  001402                  MOV     SAVER5,R5       ;DISPLAY SAVED R5
013470  004737  013522                  JSR     PC,4$           ;PRINT INITIAL MESSAGE INFO
013474  004737  013604                  JSR     PC,8$           ;PRINT SAVED VALUES
013500  005037  001400                  CLR     SAVER1          ;CLEAR SAVED REGISTER INDICATORS
013504  005037  001402                  CLR     SAVER5          ;CLEAR THE OTHER ONE
013510  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
013512  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
013514  004737  013604                  JSR     PC,8$           ;PRINT REMAINDER OF MESSAGE
013520  000207                  3$:     RTS     PC              ;RETURN
013522  105737  001376          4$:     TSTB    FRSTER          ;FIRST ERROR ?
013526  100425                          BMI     7$              ;BR IF NOT
013530  001013                          BNE     5$              ;BR IF FIRST ERROR AND PROCESSING 'DCK' ERROR
013532  004737  020176                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
013536  104414  047543                  DISPLY  ,EM42           ;DATA COMPARE ERROR
013542  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
013546  004737  020656                  JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
013552  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
013556  000404                          BR      6$              ;GO TO TYPE HEADER
013560  104414  052126          5$:     DISPLY  ,LIN9B          ;HEADER MESSAGE OF PROCESSING 'DCK' ERROR
013564  104414  001201                  DISPLY  ,$CRLF          ;CR-LF
013570  104414  052155          6$:     DISPLY  ,LIN9H          ;DISPLAY HEADER
013574  012737  177777  001376          MOV     #-1,FRSTER      ;SET ERROR FLAG
```

```
013602  000207                          7$:     RTS     PC              ;RETURN
013604  005737  001406                  8$:     TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
013610  001403                                  BEQ     9$              ;BR IF IT HAS
013612  005337  001406                          DEC     LIMIT           ;DECREMENT LIMIT COUNTER
013616  001005                                  BNE     10$             ;BR IF NOT AT LIMIT
013620  032777  000200  165326          9$:     BIT     #SW07,@SWR      ;PRINT ALL DATA COMPARE ERRORS ?
013626  001001                                  BNE     10$             ;BR IF YES
013630  000207                                  RTS     PC              ;RETURN
013632  010146                          10$:    MOV     R1,-(SP)        ;BUFFER ADDRESS
013634  162716  000002                          SUB     #2,(SP)         ;ADJUST ADDRESS
013640  004737  022210                          JSR     PC,LINOCT       ;TYPE IT
013644  104414  052716                          DISPLY  ,LINSP          ;2 SPACES
013650  016546  177776                          MOV     -2(R5),-(SP)    ;PUT GOOD DATA ON THE STACK
013654  004737  022210                          JSR     PC,LINOCT       ;TYPE IT
013660  104414  052716                          DISPLY  ,LINSP          ;2 SPACES
013664  016146  177776                          MOV     -2(R1),-(SP)    ;BAD DATA
013670  004737  022210                          JSR     PC,LINOCT       ;TYPE IT
013674  104414  001201                          DISPLY  ,$CRLF          ;CR-LF
013700  000207                                  RTS     PC              ;RETURN

                                        ;LAST LINE OF COMPARE ERROR REPORTING

013702  105737  001377                  ENDCMP: TSTB    FRSTER+1        ;ANY COMPARE ERRORS FOUND ?
013706  001417                                  BEQ     2$              ;BR IF NOT
013710  005737  001404                          TST     ERCTR           ;SEE HOW MANY ERRORS
013714  001410                                  BEQ     1$              ;BR IF ONLY CAN'T MATCH PATTERN
013716  104414  052253                          DISPLY  ,LIN9E          ;'NUMBER OF ERRORS='
013722  013746  001404                          MOV     ERCTR,-(SP)     ;NUMBER OF ERRORS
013726  004737  022242                          JSR     PC,LINDEC       ;TYPE IT
013732  104414  001201                          DISPLY  ,$CRLF          ;CR-LF
013736  004737  023424                  1$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
013742  004737  021730                          JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
013746  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

013750  010146                          MATCH:  MOV     R1,-(SP)        ;SAVE R1 ON THE STACK
013752  012704  000044                          MOV     #44,R4          ;PATTERN TABLE INDEX
013756  011601                          1$:     MOV     (SP),R1         ;RELOAD R1
013760  162704  000002                          SUB     #2,R4           ;DECREMENT INDEX
013764  001413                                  BEQ     3$              ;BR IF  PATTERN NOT MATCH
013766  016405  002476                          MOV     STNDAT(R4),R5   ;ADDRESS OF PATTERN ADDRESS
013772  012703  000004                          MOV     #4,R3           ;NUMBER OF LOCATIONS TO CHECK
013776  022125                          2$:     CMP     (R1)+,(R5)+     ;COMPARE THE BUFFER AGAINST THE PATTERN
014000  001366                                  BNE     1$              ;BR IF NOT EQUAL, TRY NEXT PATTERN
014002  005303                                  DEC     R3              ;FINISHED CHECKING?
014004  001374                                  BNE     2$              ;BR IF NOT FINISHED
014006  062704  002476                          ADD     #STNDAT,R4      ;MAKE PATTERN ADDRESS ABSOLUTE
014012  000403                                  BR      4$              ;EXIT
014014  062766  000002  000002          3$:     ADD     #2,2(SP)        ;INCREMENT RETURN ADDRESS
```

```
014022  012601                  4$:     MOV     (SP)+,R1        ;RESTORE R1
014024  000207                          RTS     PC              ;RETURN

                                ;USE ECC TO CORRECT THE DATA ERROR

014026  016037  000240  001420  ECC:     MOV     $RMBA(R0),ECSEC ;ADDRESS OF LAST LOCN XFERED
014034  016046  000236                  MOV     $RMWC(R0),-(SP) ;ACT WORDS XFERED (2'S COMP)
014040  066016  000020                  ADD     $WRDL(R0),(SP)  ;ADD WORDS REQUESTED
014044  005046                          CLR     -(SP)           ;CLEAR NEXT STACK LOCN
014046  016046  000022                  MOV     $SSEC(R0),-(SP) ;SECTOR SIZE
014052  004737  027332                  JSR     PC,LINKDV       ;DIVIDE WORDS XFERED BY SECTOR SIZE
014056  005716                          TST     (SP)            ;PARTIAL SECTOR XFERED ?
014060  001413                          BEQ     1$              ;BR IF NOT
014062  006316                          ASL     (SP)            ;CONVERT INTO NUMBER OF BYTES
014064  161637  001420                  SUB     (SP),ECSEC      ;SUBTRACT SECTOR RESIDUE
014070  126027  000024  000005          CMPB    $CODE(R0),#5    ;WAS OP READ HEAD & DATA
014076  001007                          BNE     2$              ;BR IF NOT
014100  062737  000004  001420          ADD     #4,ECSEC        ;ADD HEADER SIZE (IN BYTES) BACK IN
014106  000403                          BR      2$              ;GO ADJUST THE STACK POINTER
014110  162737  001000  001420  1$:     SUB     #1000,ECSEC     ;SUBTRACT SECTOR DATA FIELD SIZE (IN BYTES)
014116  062706  000004          2$:     ADD     #4,SP           ;ADJUST THE STACK POINTER
014122  016037  000300  001416          MOV     $RMEC1(R0),ECBIT  ;ECC POSITION COUNT
014130  005337  001416                  DEC     ECBIT           ;ADJUST THE POSITION COUNT
014134  013737  001416  001426          MOV     ECBIT,ECWRD     ;LOAD THE WORD COUNT LOCATION
014142  042737  177760  001416          BIC     #^C17,ECBIT     ;SAVE THE BIT OFFSET COUNT
014150  042737  000017  001426          BIC     #17,ECWRD       ;CLEAR THE BIT OFFSET
014156  006237  001426                  ASR     ECWRD           ;CHANGE TO BYTE COUNT
014162  006237  001426                  ASR     ECWRD           ;CHANGE TO BYTE COUNT
014166  006237  001426                  ASR     ECWRD           ;CHANGE TO BYTE COUNT
014172  104414  052332                  DISPLY  ,LIN10A         ;'ERROR BURST BEGINS AT '
014176  013746  001426                  MOV     ECWRD,-(SP)     ;PUT THE WORD COUNT ON THE STACK
014202  006216                          ASR     (SP)            ;CONVERT TO WORD COUNT FOR MESSAGE
014204  004737  030246                  JSR     PC,$SB2D        ;CONVERT THE WORD COUNT
014210  004737  027646                  JSR     PC,$SUPRS       ;PRINT IT
014214  104414  052366                  DISPLY  ,LIN10B         ;' IN DATA FIELD OF ERROR SECTOR'
014220  063737  001420  001426          ADD     ECSEC,ECWRD     ;FIND THE BEGINNING OF THE ERROR BURST
014226  026037  000240  001426          CMP     $RMBA(R0),ECWRD  ;SEE IF BURST WAS IN DATA READ
014234  101002                          BHI     .+6             ;BR IF IN DATA READ
014236  000137  014554                  JMP     ECC2            ;NOT IN DATA READ - REPORT IT
014242  016037  000302  001422          MOV     $RMEC2(R0),ECMSK0  ;GET THE ERROR MASK
014250  005037  001424                  CLR     ECMSK1          ;CLEAR THE UPPER MASK WORD
014254  005737  001416          3$:     TST     ECBIT           ;BIT OFFSET EQUAL ZERO
014260  001407                          BEQ     4$              ;BR IF IT IS
014262  005337  001416                  DEC     ECBIT           ;DECREMENT THE BIT OFFSET COUNT
014266  006337  001422                  ASL     ECMSK0          ;SHIFT THE ERROR MASK
014272  006137  001424                  ROL     ECMSK1          ;SHIFT THE LOWER INTO THE UPPER
014276  000766                          BR      3$              ;CONTINUE THE SHIFT
014300  017737  165122  001432  4$:     MOV     @ECWRD,ECBAD0   ;SAVE THE INCORRECT WORD
014306  005037  001434                  CLR     ECWRD1          ;CLEAR SECOND INCORRECT WORD ADDRESS
014312  013746  001422                  MOV     ECMSK0,-(SP)    ;PUT LOWER MASK ON STACK
014316  047716  165104                  BIC     @ECWRD,(SP)     ;CLEAR ERRONEOUS ONE BITS FROM MASK
014322  043777  001422  165076          BIC     ECMSK0,@ECWRD   ;CLEAR ERRONEOUS ONE BITS FROM BAD WORD
014330  052677  165072                  BIS     (SP)+,@ECWRD    ;SET DROPPED BITS
014334  005737  001424                  TST     ECMSK1          ;DOES BURST GO INTO NEXT WORD ?
014340  001431                          BEQ     ECC1            ;BR IF BURST ONLY IN ONE WORD
014342  013737  001426  001434          MOV     ECWRD,ECWRD1    ;DUPLICATE ADDRESS
```

```
014350  062737  000002  001434          ADD     #2,ECWRD1          ;INCREMENT ERROR ADDRESS
014356  026037  000240  001434          CMP     $RMBA(R0),ECWRD1   ;IS NEXT WORD IN THE BUFFER
014364  101003                          BHI     5$                 ;BR IF IT IS
014366  005037  001434                  CLR     ECWRD1             ;CLEAR 2ND WORD ADDRESS
014372  000414                          BR      ECC1               ;PRINT WORD CORRECTED
014374  017737  165034  001440  5$:     MOV     @ECWRD1,ECBAD1     ;SAVE THE SECOND BAD WORD
014402  013746  001424                  MOV     ECMSK1,-(SP)       ;PUT THE UPPER MASK ON THE STACK
014406  047716  165022                  BIC     @ECWRD1,(SP)       ;CLEAR ERRONEOUS ONE BITS FROM UPPER MASK
014412  043777  001424  165014          BIC     ECMSK1,@ECWRD1     ;CLEAR ERRONEOUS ONE BITS FROM DATA WORD
014420  052677  165010                  BIS     (SP)+,@ECWRD1      ;SET DROPPED BITS
014424  104414  052534          ECC1:   DISPLY  ,LIN10H            ;HEADER
014430  013746  001426                  MOV     ECWRD,-(SP)        ;PUT ECWRD ON THE STACK
014434  004737  022210                  JSR     PC,LINOCT          ;TYPE ECWRD
014440  104414  052716                  DISPLY  ,LINSP             ;SPACES
014444  013746  001432                  MOV     ECBADO,-(SP)       ;PUT ECBADO ON THE STACK
014450  004737  022210                  JSR     PC,LINOCT          ;TYPE ECBADO
014454  104414  052716                  DISPLY  ,LINSP             ;SPACES
014460  017746  164742                  MOV     @ECWRD,-(SP)       ;PUT @ECWRD ON THE STACK
014464  004737  022210                  JSR     PC,LINOCT          ;TYPE @ECWRD
014470  104414  052716                  DISPLY  ,LINSP             ;SPACES
014474  005737  001434                  TST     ECWRD1             ;PRINT THE NEXT WORD ?
014500  001427                          BEQ     ECCX               ;BR IF NOT
014502  104414  001201                  DISPLY  ,$CRLF             ;CR-LF
014506  013746  001434                  MOV     ECWRD1,-(SP)       ;PUT ECWRD1 ON THE STACK
014512  004737  022210                  JSR     PC,LINOCT          ;TYPE ECWRD1
014516  104414  052716                  DISPLY  ,LINSP             ;SPACES
014522  013746  001440                  MOV     ECBAD1,-(SP)       ;PUT ECBAD1 ON THE STACK
014526  004737  022210                  JSR     PC,LINOCT          ;TYPE ECBAD1
014532  104414  052716                  DISPLY  ,LINSP             ;SPACES
014536  017746  164672                  MOV     @ECWRD1,-(SP)      ;PUT @ECWRD1 ON THE STACK
014542  004737  022210                  JSR     PC,LINOCT          ;TYPE @ECWRD1
014546  104414  052716                  DISPLY  ,LINSP             ;SPACES
014552  000402                          BR      ECCX               ;EXIT
014554  104414  052427          ECC2:   DISPLY  ,LIN10C            ;ERROR BURST WAS NOT TRANSFERED TO MEMORY
014560  104414  001201          ECCX:   DISPLY  ,$CRLF             ;CR-LF
014564  000207                          RTS     PC                 ;RETURN

                                ;ROUTINE TO DISPLAY THE SECTOR WHICH GAVE THE HARD ERROR

014566  032777  000010  164360  PRTBAD: BIT     #SW3,@SWR          ;PRINT THE BAD SECTOR ?
014574  001460                          BEQ     6$                 ;BR IF NOT
014576  016001  000240                  MOV     $RMBA(R0),R1       ;PUT THE END ADDRESS INTO R1
014602  016046  000020                  MOV     $WRDL(R0),-(SP)    ;FIND THE BEGINNING OF THE SECTOR
014606  066016  000236                  ADD     $RMWC(R0),(SP)     ;SUBTRACT THE WORDS NOT TRANSFERED
014612  005046                          CLR     -(SP)              ;MAKE THE UPPER DIVIDEND 0
014614  016046  000022                  MOV     $SSEC(R0),-(SP)    ;DIVDE THE WORDS TRANSFERED BY THE SECTOR SIZE
014620  004737  027332                  JSR     PC,LINKDV          ;DIVIDE
014624  005716                          TST     (SP)               ;REMANDER = 0 ?
014626  001403                          BEQ     1$                 ;BR IF IT IS - COMPLETE SECTOR TRANSFERED
014630  006316                          ASL     (SP)               ;CONVERT THE RESIDUAL SECTOR SIZE INTO BYTE COUNT
014632  161601                          SUB     (SP),R1            ;SUBTRACT IT FROM THE END ADDRESS
014634  000410                          BR      2$                 ;FINISH THE SIZING
014636  162701  001000          1$:     SUB     #1000,R1           ;SUBTRACT FULL SECTOR SIZE FROM END ADDR
014642  126027  000024  000005          CMPB    $CODE(R0),#5       ;WAS OPERATION READ HEADER & DATA ?
014650  001002                          BNE     2$                 ;BR IF NOT
014652  162701  000004                  SUB     #4,R1              ;SUBTRACK HEADER SIZE FROM ADDR
```

```
014656  062706  000004          2$:     ADD     #4,SP           ;RESTORE THE STACK POINTER
014662  104414  052621                  DISPLY  ,LIN11H         ;PRINT THE HEADER
014666  012702  000007          3$:     MOV     #7,R2           ;R2 CONTAINS THE WORDS/LINE COUNT
014672  010146                          MOV     R1,-(SP)        ;PUT THE ADDRESS ON THE STACK
014674  004737  022210                  JSR     PC,LINOCT       ;TYPE THE ADDRESS
014700  020160  000240          4$:     CMP     R1,$RMBA(R0)    ;PRINTED ALL THE SECTOR ?
014704  001412                          BEQ     5$              ;BR IF ALL PRINTED
014706  104414  052716                  DISPLY  ,LINSP          ;SPACES
014712  012146                          MOV     (R1)+,-(SP)     ;PUT THE DATA ON THE STACK
014714  004737  022210                  JSR     PC,LINOCT       ;TYPE THE DATA
014720  005302                          DEC     R2              ;DECREMENT THE HORIZONTAL COUNT
014722  001366                          BNE     4$              ;BR IF NOT AT THE END OF THE LINE
014724  104414  001201                  DISPLY  ,$CRLF          ;CR-LF
014730  000756                          BR      3$              ;RESTORE THE WORDS/LINE COUNT
014732  104414  001201          5$:     DISPLY  ,$CRLF          ;PRINT WHAT REMAINS IN THE BUFFER
014736  000207                  6$:     RTS     PC              ;RETURN

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

014740  111037  045532          RTNCTR: MOVB    (R0),GENDPB     ;MOVE THE DRIVE # TO THE GENERAL DPB
014744  112737  000117  045534          MOVB    #RTC,GENDPB+$COMND  ;COMMAND CODE
014752  004037  035040          1$:     JSR     R0,RM03         ;DRIVER ENTRANCE
014756  045532                          GENDPB                  ;DPB ADDRESS FOR ORDER
014760  000774                          BR      1$              ;DRIVER DIDN'T ACCEPT ORDER
014762  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,RECALTO
                                ;       RETURN

014764  111037  045532          RECALT: MOVB    (R0),GENDPB     ;MOVE THE DRIVE # TO THE GENERAL DPB
014770  112737  000107  045534  RECALO: MOVB    #RECAL,GENDPB+$COMND  ;RELCALIBRATE COMMAND
014776  004037  035040          1$:     JSR     R0,RM03         ;DRIVER ENTRANCE
015002  045532                          GENDPB                  ;DPB ADDRESS FOR ORDER
015004  000774                          BR      1$              ;DRIVER DIDN'T ACCEPT THE ORDER
015006  005737  045550          2$:     TST     GENDPB+$STATUS  ;SEE IF FINISHED
015012  001775                          BEQ     2$              ;BR IF NOT FINISHED
015014  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

015016  111037  045532                  OFFST:  MOVB    (R0),GENDPB     ;DRIVE # TO GENERAL DPB
015[illegible]2  112737  000115  045534         MOVB    #OFFSET,GENDPB+$COMMAND ;COMMAND
015[illegible]0  004[illegible]37  035040        1$:     JSR     R0,RM03         ;DRIVER ENTRANCE
015[illegible]34  0[illegible]32                         GENDPB                  ;DPB ADDRESS FOR ORDER
015[illegible]36  000774                                 BR      1$              ;DRIVER DIDN'T ACCEPT ORDER
015040  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,READOR
                                        ;       RETURN

015042  116637  000004  045543          READHD: MOVB    4(SP),GENDPB+$TRK ;TRACK ADDRESS
015050  116637  000006  045542                  MOVB    6(SP),GENDPB+$SEC ;SECTOR ADDRESS
015056  016637  000002  045544                  MOV     2(SP),GENDPB+$CYL       ;CYLINDER ADDRESS
015064  111037  045532                          MOVB    (R0),GENDPB     ;DRIVE NUMBER
015070  112737  000173  045534                  MOVB    #RDHD,GENDPB+$COMMAND ;COMMAND
015076  012737  177776  045536                  MOV     #-2,GENDPB+$WRDM        ;WORD CTR = 2
015104  004037  035040          1$:             JSR     R0,RM03         ;DRIVER ENTRANCE
015110  045532                                  GENDPB                  ;DPB ADDRESS FOR ORDER
015112  000774                                  BR      1$              ;DRIVER DIDN'T ACCEPT COMMAND
015114  005737  045550          2$:             TST     GENDPB+$STATUS  ;FINISHED?
015120  001775                                  BEQ     2$              ;BR IF NOT
015122  011666  000006                          MOV     (SP),6(SP)      ;ADJUST STACK FOR RETURN
015126  062706  000006                          ADD     #6,SP           ;ADJUST RETRUN POINTER
015132  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'

015134  004737  016132          $RETRY: JSR     PC,GODRIV       ;RE-START ORDER
015140  005760  000016          1$:     TST     STATUS(R0)      ;ORDER FINISHED?
015144  001775                          BEQ     1$              ;BR IF NOT
015146  100405                          BMI     2$              ;BR IF ERROR
015150  105237  001351                  INCB    RETRY+1         ;INCREMENT RETRY COUNT
015154  062716  000002                  ADD     #2,(SP)         ;INCREMENT RETURN
015160  000425                          BR      5$              ;GO TO EXIT
015162  032760  000200  000016  2$:     BIT     #BIT7,STATUS(R0)  ;DID ORDER TERMINATE NORMALLY ?
015170  001430                          BEQ     7$              ;BR IF NOT
015172  005737  001346                  TST     MASK            ;IS ERROR MASK 0 ?
015176  001004                          BNE     3$              ;BR IF NOT
015200  005760  000250                  TST     $RMER1(R0)      ;MAKE SURE THAT THE DRIVE ERROR REG IS CLEAR
015204  001014                          BNE     6$              ;BR IF NOT
```

```
015206  000404                          BR      4$                  ;CONTINUE RETRY
015210  033760  001346  000250  3$:     BIT     MASK,$RMER1(R0)     ;SAME ERROR?
015216  001407                          BEQ     6$                  ;BR IF NOT
015220  105237  001351          4$:     INCB    RETRY+1             ;INCREMENT RETRY COUNT
015224  123737  001350  001351          CMPB    RETRY,RETRY+1       ;DONE ?
015232  001340                          BNE     $RETRY              ;BR IF NOT DONE
015234  000207                  5$:     RTS     PC                  ;RETURN
015236  004737  022176          6$:     JSR     PC,LINE8            ;REPORT DIFFERENT ERROR
015242  004737  021730                  JSR     PC,LINE7            ;PRINT LINE 7
015246  005726                          TST     (SP)+               ;ADJUST STACK POINTER FOR DIRECT RETURN
015250  000207                          RTS     PC                  ;RETURN
015252  104414  052103          7$:     DISPLY  ,LIN8M              ;'DIFFERENT ERROR DURING RETRY'
015256  000137  006364                  JMP     ERPRC1              ;REPORT THE ERROR

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

015262  032760  000300  000016  STATIS: BIT     #BIT07!BIT06,$STATUS(R0) ;CHECK FOR DATA TERMINATION
015270  001451                          BEQ     3$                  ;BR IF NOT DATA TERMINATION
015272  016037  000240  015416          MOV     $RMBA(R0),FACTOR    ;STORE THE FINAL BUFFER ADDRESS
015300  166037  000206  015416          SUB     $BUF(R0),FACTOR     ;SUBTRACT THE INITIAL ADDRESS
015306  001431                          BEQ     2$                  ;BR IF NO DATA TRANSFER
015310  006237  015416                  ASR     FACTOR              ;CONVERT TO A WORD COUNT
015314  063760  015416  000046          ADD     FACTOR,$TRANS(R0)   ;UPDATE WORD COUNT
015322  005560  000050                  ADC     $TRANS+2(R0)        ;ADD ANY CARRY
015326  132760  000002  000024          BITB    #BIT01,$CODE(R0)    ;SEE IF ORDER READ OR WRITE
015334  001016                          BNE     2$                  ;BRANCH IF ORDER WRITE
015336  126027  000024  000001          CMPB    $CODE(R0),#1        ;PRESENT OPERATION AN AUTOMATIC WRITE CHECK ?
015344  101005                          BHI     1$                  ;BR IF NOT
015346  066060  000020  000046          ADD     $WRDL(R0),$TRANS(R0)  ;ADD WORDS WRITTEN
015354  005560  000050                  ADC     $TRANS+2(R0)        ;ADD A CARRY
015360  063760  015416  000052  1$:     ADD     FACTOR,$READ(R0)    ;UPDATE THE READ WORD COUNT
015366  005560  000054                  ADC     $READ+2(R0)         ;ADD ANY CARRY
015372  026060  000012  000270  2$:     CMP     $CYL(R0),$RMDC(R0)  ;DID MID-TRANSFER SEEK OCCUR
015400  001405                          BEQ     3$                  ;BR IF NOT
015402  062760  000001  000042          ADD     #1,$POSIT(R0)       ;INCREMENT SEEK COUNT
015410  005560  000044                  ADC     $POSIT+2(R0)        ;ADD CARRY TO UPPER WORD
015414  000207                  3$:     RTS     PC

015416  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

015420  010146                  GETBUF: MOV     R1,-(SP)            ;SAVE R1
015422  010246                          MOV     R2,-(SP)            ;SAVE R2
015424  010346                          MOV     R3,-(SP)            ;SAVE R3
015426  013702  001700                  MOV     BUFTBL,R2           ;NUMBER OF SEPARATE BUFFERS
```

```
015432  001444                          BEQ     6$              ;BR IF NONE AVAILABLE
015434  012701  001702                  MOV     #BUFTBL+2,R1    ;FIRST ADDRESS OF ALLOCATION TABLE
015440  026061  000020  000002  1$:     CMP     $WRDL(R0),2(R1) ;SEE IF THERE IS A BLOCK LARGE ENOUGH
015446  101405                          BLOS    3$              ;BRANCH IF IT IS
015450  005302                          DEC     R2              ;DECREMENT TABLE COUNT
015452  001434                          BEQ     6$              ;BR IF THROUGH TABLE
015454  062701  000004                  ADD     #4,R1           ;INCREMENT TABLE POINTER
015460  000767                          BR      1$              ;CONTINUE LOOKING
015462  011166  000010          3$:     MOV     (R1),10(SP)     ;BUFFER ADDRESS TO STACK
015466  166061  000020  000002          SUB     $WRDL(R0),2(R1) ;ADJUST BUFFER SIZE
015474  001407                          BEQ     4$              ;BR IF DIFFERENCE IS ZERO
015476  006360  000020                  ASL     $WRDL(R0)       ;CONVERT # WORDS TO BYTES
015502  066011  000020                  ADD     $WRDL(R0),(R1)  ;MAKE NEW STARTING ADDRESS
015506  006260  000020                  ASR     $WRDL(R0)       ;RETURN # BYTES TO WORDS
015512  000414                          BR      6$              ;RETURN
015514  005337  001700          4$:     DEC     BUFTBL          ;DECREMENT ENTRIES COUNT
015520  001411                          BEQ     6$              ;BR IF ALLOCATION TABLE EMPTY
015522  005302                          DEC     R2              ;DECREMENT TABLE COUNT
015524  001407                          BEQ     6$              ;BR IF ITEM WERE LAST ENTRY
015526  010103                          MOV     R1,R3           ;MOVE TABLE POINTER
015530  062703  000004                  ADD     #4,R3           ;POINT TO NEXT ENTRY
015534  012321                  5$:     MOV     (R3)+,(R1)+     ;MOVE ITEMS
015536  012321                          MOV     (R3)+,(R1)+
015540  005302                          DEC     R2              ;DECREMENT TABLE COUNT
015542  001374                          BNE     5$              ;CONTINUE IF NOT AT END OF TABLE
015544  012603                  6$:     MOV     (SP)+,R3        ;RESTORE R3
015546  012602                          MOV     (SP)+,R2        ;RESTORE R2
015550  012601                          MOV     (SP)+,R1        ;RESTORE R1
015552  000207                          RTS     PC              ;RETURN


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

015554  010146                  RELBUF: MOV     R1,-(SP)        ;SAVE R1
015556  010246                          MOV     R2,-(SP)        ;SAVE R2
015560  010446                          MOV     R4,-(SP)        ;SAVE R4
015562  010546                          MOV     R5,-(SP)        ;SAVE R5
015564  012701  001702                  MOV     #BUFTBL+2,R1    ;BEGINNING OF TABLE
015570  013702  001700                  MOV     BUFTBL,R2       ;ENTRY COUNT
015574  001424                          BEQ     2$              ;BR IF EMPTY TABLE
015576  016003  000020                  MOV     $WRDL(R0),R3    ;TRIAL ADDRESS
015602  006303                          ASL     R3              ;CHANGE TO BYTE COUNT
015604  066003  000006                  ADD     $BUF(R0),R3     ;ADDRESS OF HIGHER ADJACENT BLOCK
015610  021103                  1$:     CMP     (R1),R3         ;UPPER ADJACENT BLOCK
015612  001424                          BEQ     4$              ;BR IF YES
015614  062701  000004                  ADD     #4,R1           ;INCREMENT POINTER
015620  005302                          DEC     R2              ;DECREMENT ENTRY COUNT
015622  001372                          BNE     1$              ;CONTINUE SEARCHING
015624  016011  000006                  MOV     $BUF(R0),(R1)   ;PUT THE BUFFER BLOCK INTO THE TABLE
015630  016061  000020  000002          MOV     $WRDL(R0),2(R1)  ;BLOCK SIZE
015636  005237  001700                  INC     BUFTBL          ;INCREMENT ENTRY COUNT
015642  005202                          INC     R2              ;INCREMENT R2 FOR USE LATER
```

```
015644  000414                          BR      5$              ;SEE IF A LOWER ADJACENT BLOCK IS IN THE TABLE
015646  016021  000006          2$:     MOV     $BUF(R0),(R1)+  ;BLOCK ADDRESS TO TABLE
015652  016021  000020                  MOV     $WRDL(R0),(R1)+ ;SIZE TO TABLE
015656  005237  001700                  INC     BUFTBL          ;INCREMENT ENTRY COUNT
015662  000443                          BR      10$             ;EXIT
015664  016011  000006          4$:     MOV     $BUF(R0),(R1)   ;RELEASED BUFFER IS LOWER ADJACENT
015670  066061  000020  000002          ADD     $WRDL(R0),2(R1) ;INCREMENTED SIZE
015676  010246                  5$:     MOV     R2,-(SP)        ;SAVE R2
015700  013702  001700                  MOV     BUFTBL,R2       ;ENTRY COUNT
015704  012705  001702                  MOV     #BUFTBL+2,R5    ;BEGINNING OF TABLE
015710  016504  000002          6$:     MOV     2(R5),R4        ;BLOCK SIZE (IN WORDS)
015714  006304                          ASL     R4              ;CHANGE TO BYTE COUNT
015716  061504                          ADD     (R5),R4         ;ADD BLOCK BEGINNING ADDRESS
015720  020411                          CMP     R4,(R1)         ;R1 STILL POINTS TO INSERTED ENTRY
015722  001406                          BEQ     8$              ;LOWER ADJACENT IN TABLE
015724  062705  000004                  ADD     #4,R5           ;INCREMENT POINTER
015730  005302                          DEC     R2              ;DECREMENT ENTRY COUNT
015732  001366                          BNE     6$              ;CONTINUE LOOKING
015734  005726                          TST     (SP)+           ;RESTORE STACK POINTER
015736  000415                          BR      10$             ;END
015740  012602                  8$:     MOV     (SP)+,R2        ;RESTORE R2
015742  066165  000002  000002          ADD     2(R1),2(R5)     ;INCREMENT LOWER BLOCK LENGTH
015750  005337  001700                  DEC     BUFTBL          ;DECREMENT ENTRY COUNT
015754  010105                          MOV     R1,R5           ;GET READY TO COMPRESS
015756  062705  000004                  ADD     #4,R5           ;INCREMENT TO NEXT ENTRY
015762  012521                  9$:     MOV     (R5)+,(R1)+     ;COMPRESS TABLE
015764  012521                          MOV     (R5)+,(R1)+     ;MOVE SIZE FIELD DOWN
015766  005302                          DEC     R2              ;DECREMENT ENTRY COUNT
015770  001374                          BNE     9$              ;BR IF NOT FINISHED
015772  012605                  10$:    MOV     (SP)+,R5        ;RESTORE R5
015774  012604                          MOV     (SP)+,R4        ;RESTORE R4
015776  012602                          MOV     (SP)+,R2        ;RESTORE R2
016000  012601                          MOV     (SP)+,R1        ;RESTORE R1
016002  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

016004  104412                  FILBUF: SAVREG                  ;SAVE THE REGISTERS
016006  132760  000004  000024          BITB    #BIT02,$CODE(R0)  ;SEE IF READ ORDER
016014  001044                          BNE     4$              ;BR IF READ
016016  016001  000006          1$:     MOV     $BUF(R0),R1     ;BUFFER ADDRESS
016022  016002  000020                  MOV     $WRDL(R0),R2    ;POSITIVE WORD COUNT
016026  132760  000001  000024          BITB    #BIT00,$CODE(R0)  ;SEE IF WRITE HEADER TYPE ORDER
016034  001413                          BEQ     2$              ;BR IF NOT
016036  016011  000012                  MOV     $CYL(R0),(R1)   ;CYLINDER ADDRESS
016042  052711  010000                  BIS     #BIT12,(R1)     ;SET FMT22 BIT
016046  052721  140000                  BIS     #140000,(R1)+   ;SET MFG AND USER BITS
016052  016021  000010                  MOV     $SEC(R0),(R1)+  ;MOVE SECTOR & TRACK
016056  162702  000002                  SUB     #2,R2           ;ADJUST THE WORD COUNT
```

```
016062  003421                          BLE     4$              ;BR IF END OF PATTERN
016064  005004                  2$:     CLR     R4              ;CLEAR R4
016066  116004  000030                  MOVB    $PATTC(R0),R4   ;RELATIVE PATTERN ADDRESS
016072  016405  002476                  MOV     STNDAT(R4),R5   ;PATTERN ADDRESS
016076  012703  000020                  MOV     #20,R3          ;PATTERN COUNT
016102  012521                  3$:     MOV     (R5)+,(R1)+     ;MOVE THE PATTERN INTO THE BUFFER
016104  005302                          DEC     R2              ;DECREMENT THE WORD COUNT
016106  003407                          BLE     4$              ;BR IF DONE (WORD COUNT = 0)
016110  005303                          DEC     R3              ;DECREMENT THE PATTERN COUNT
016112  001373                          BNE     3$              ;BR IF MORE PATTERN
016114  012703  000020                  MOV     #20,R3          ;RESTORE PATTERN COUNT
016120  016405  002476                  MOV     STNDAT(R4),R5   ;RESTORE THE ADDRESS
016124  000766                          BR      3$              ;CONTINUE DISTRIBUTING THE PATTERN
016126  104413                  4$:     RESREG                  ;RESTORE THE REGISTERS
016130  000207                          RTS     PC              ;RETURN

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

016132  010046                  GODRIV: MOV     R0,-(SP)        ;SAVE R0
016134  010037  016144                  MOV     R0,2$           ;CURRENT DPB ADDRESS
016140  004037  035040          1$:     JSR     R0,RM03         ;CALL THE DRIVE HANDLER
016144  000000                  2$:     .WORD   0               ;DRIVE BLOCK ADDRESS GOES HERE
016146  000000                          HALT                    ;DRIVER REJECTED REQUEST
016150  012600                          MOV     (SP)+,R0        ;RESTORE R0
016152  062760  000001  000036          ADD     #1,$OPERC(R0)   ;INCREMENT THE OPERATION COUNT
016160  005560  000040                  ADC     $OPERC+2(R0)
016164  026060  000034  000012          CMP     $PREVA+2(R0),$CYL(R0)  ;DID ORDER REQUIRE A CYLINDER CHANGE
016172  001405                          BEQ     3$              ;BR IF NOT
016174  062760  000001  000042          ADD     #1,$POSIT(R0)   ;INCREMENT SEEK COUNT
016202  005560  000044                  ADC     $POSIT+2(R0)    ;ADD ANY CARRY
016206  000207                  3$:     RTS     PC

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

016210  004737  033350          SELPAR: JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
016214  032777  000001  162732          BIT     #SW0,@SWR       ;SEE IF SW0 SET
016222  001012                          BNE     2$              ;BR IF SET - READ ONLY
016224  012705  000010          1$:     MOV     #10,R5          ;READ/WRITE SELECTION DIVISOR
016230  004737  027304                  JSR     PC,GETREM       ;GET SELECTION VALUE
016234  020537  001500                  CMP     R5,RATIO        ;DETERMINE IF READ OR WRITE
016240  103003                          BHIS    2$              ;BR IF READ
016242  004737  017066                  JSR     PC,RANWRT       ;SELECT A WRITE ORDER
016246  000410                          BR      THEAD           ;SELECT ADDRESS
016250  013705  033450          2$:     MOV     $LONUM,R5       ;SELECT READ OPERATION CODE
016254  042705  177776                  BIC     #^C1,R5         ;MASK OUT ALL BUT BIT 0
016260  062705  000004                  ADD     #4,R5           ;TABLE OFFSET FOR READ CODE
016264  110560  000074                  MOVB    R5,$NCODE(R0)   ;ORDER SELECTION CODE TO CONTROL BLOCK
016270  005737  001512          THEAD:  TST     HEADER  ;ENABLE RANDOM ADDRESS SELECT ?
```

```
016274  001425                          BEQ     RANSEC          ;YES
016276  016060  000242  000076          MOV     $RMDA(R0),$NSEC(R0)     ;SECTOR AND TRACK
016304  016060  000270  000100          MOV     $RMDC(R0),$NCYL(R0) ;CYLINDER NUMBER
016312  026060  000100  000106          CMP     $NCYL(R0),MAXCYL(R0) ;OVER MAX CYLINDER # ?
016320  103520                          BLO     XWCNT           ;NO
016322  016060  000110  000100          MOV     MINCYL(R0),$NCYL(R0) ;RESET CYLINDER NUMBER
016330  116060  000120  000076          MOVB    MINSEC(R0),$NSEC(R0) ;RESET SECTOR NUMBER
016336  116060  000114  000077          MOVB    MINTRK(R0),$NTRK(R0) ;RESET TRACK NUMBER
016344  000137  016562                  JMP     XWCNT           ;SELECT BUFFER SIZE

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

016350  016005  000116          RANSEC: MOV     MAXSEC(R0),R5   ;GET MAXIMUM SECTOR ADDRESS
016354  026005  000120                  CMP     MINSEC(R0),R5   ;'MINSEC' AND 'MAXSEC' THE SAME ?
016360  001417                          BEQ     2$              ;BR IF THEY ARE
016362  166005  000120                  SUB     MINSEC(R0),R5   ;SUBTRACT MINIMUM SECTOR ADDRESS
016366  100002                          BPL     1$              ;BR IF MAX LARGER THAN MIN
016370  062705  000040                  ADD     #32.,R5         ;CORRECT THE NUMBER
016374  005205                  1$:     INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
016376  004737  027304                  JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
016402  066005  000120                  ADD     MINSEC(R0),R5   ;NEW ADDRESS
016406  020527  000037                  CMP     R5,#31.         ;IS VALUE TOO LARGE ?
016412  101402                          BLOS    2$              ;BR IF NOT
016414  162705  000040                  SUB     #32.,R5         ;CORRECT VALUE
016420  110560  000076          2$:     MOVB    R5,$NSEC(R0)    ;STORE SECTOR ADDRESS IN DPB

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

016424  016005  000112          RANTRK: MOV     MAXTRK(R0),R5   ;GET MAXIMUM TRACK ADDRESS
016430  026005  000114                  CMP     MINTRK(R0),R5   ;'MINTRK' AND 'MAXTRK' THE SAME ?
016434  001417                          BEQ     2$              ;BR IF THEY ARE
016436  166005  000114                  SUB     MINTRK(R0),R5   ;SUBTRACT MINIMUM TRACK ADDRESS
016442  100002                          BPL     1$              ;BR IF MAX LARGER THAN MIN
016444  062705  000005                  ADD     #5.,R5          ;CORRECT THE NUMBER
016450  005205                  1$:     INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
016452  004737  027304                  JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
016456  066005  000114                  ADD     MINTRK(R0),R5   ;NEW TRACK ADDRESS
016462  020527  000004                  CMP     R5,#4.          ;IS VALUE TOO LARGE ?
016466  101402                          BLOS    2$              ;BR IF NOT
016470  162705  000005                  SUB     #5.,R5          ;CORRECT VALUE
016474  110560  000077          2$:     MOVB    R5,$NTRK(R0)    ;STORE TRACK ADDRESS IN DPB

                                ;GENERATE A RANDOM CYLINDER ADDRESS BETWEEN VALUES 'MINCYL' & 'MAXCYL'

016500  016037  000106  001446          MOV     MAXCYL(R0),CYLIMT       ;ASSUME AN RM03
016506  016005  000106          RANCYL: MOV     MAXCYL(R0),R5   ;GET MAXIMUM CYLINDER ADDRESS
016512  026005  000110                  CMP     MINCYL(R0),R5   ;'MINCYL' AND 'MAXCYL' THE SAME ?
016516  001417                          BEQ     2$              ;BR IF THEY ARE
016520  166005  000110                  SUB     MINCYL(R0),R5   ;SUBTRACT MINIMUM CYLINDER ADDRESS
016524  100002                          BPL     1$              ;BR IF MAX LARGER THAN MIN
016526  063705  001446                  ADD     CYLIMT,R5       ;CORRECT THE NUMBER
016532  005205                  1$:     INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
016534  004737  027304                  JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
016540  066005  000110                  ADD     MINCYL(R0),R5   ;NEW CYLINDER ADDRESS
016544  023705  001446                  CMP     CYLIMT,R5       ;IS VALUE TOO LARGE ?
016550  003002                          BGT     2$              ;BR IF NOT
```

```
016552  016005  000110                  MOV     MINCYL(R0),R5     ;CORRECT VALUE
016556  010560  000100          2$:     MOV     R5,$NCYL(R0)      ;STORE CYLINDER ADDRESS IN DPB
016562  132760  000001  000074  XWCNT:  BITB    #BIT00,$NCODE(R0) ;HEADER OPERATION INVOLED ?
016570  001414                          BEQ     RANSIZ            ;NO
016572  012760  000402  000102          MOV     #258.,$NWRDL(R0)  ;CHANGE WORD LENGTH TO 258 FOR WRTHD ORDER
016600  122760  000005  000074  3$:     CMPB    #5,$NCODE(R0)     ;READ HEADER AND DATA ?
016606  001461                          BEQ     RANXIT            ;YES
016610  004537  016762                  JSR     R5,CHKADR         ;IF WRITE HEAD AND DATA COMMAND
                                                                  ;AVOID WRITING BAD SPOT HEADER
016614  000452                          BR      RANPAT            ;BRANCH IF NOT ON BAD SPOT
016616  000137  016210                  JMP     SELPAR            ;SELECT THE PARAMETERS AGAIN

                                ;GENERATE A RANDOM BUFFER LENGTH BETWEEN 4 & THE VALUE IN 'MAXDL'

016622  013705  001462          RANSIZ: MOV     MAXDL,R5          ;GET BUFFER SIZE
016626  005737  001476                  TST     WCSEL             ;SELECT A RANDOM WORD COUNT ?
016632  001010                          BNE     1$                ;BR IF NOT
016634  005205                          INC     R5                ;INCREMENT THE MAXIMUM SIZE
016636  004737  027304                  JSR     PC,GETREM         ;DIVIDE BY MAX VALUE
016642  005705                          TST     R5                ;IS THE REMAINDER 0 ?
016644  001003                          BNE     1$                ;NOT 0, CONTINUE
016646  004737  033350                  JSR     PC,$RAND          ;CYCLE THE RANDOM NUMBER GENERATOR
016652  000763                          BR      RANSIZ            ;TRY AGAIN
016654  022705  000004          1$:     CMP     #4,R5             ;LESS THAN 4 ?
016660  003403                          BLE     2$                ;NO
016662  012705  000004                  MOV     #4,R5             ;SET SIZE TO 4
016666  000405                          BR      3$
016670  023705  001462          2$:     CMP     MAXDL,R5          ;LARGE THAN MAX ALLOWED ?
016674  101002                          BHI     3$                ;NO
016676  013705  001462                  MOV     MAXDL,R5          ;RESET COUNTER
016702  132760  000004  000074  3$:     BITB    #BIT02,$NCODE(R0) ;READ OPERATION ?
016710  001006                          BNE     4$                ;YES
016712  042705  000377                  BIC     #377,R5           ;SECTOR BOUNDARY FOR WRITE OP
016716  005705                          TST     R5                ;NONE
016720  003002                          BGT     4$                ;NO
016722  012705  000400                  MOV     #400,R5           ;AT LEAST ONE SECTOR
016726  010560  000102          4$:     MOV     R5,$NWRDL(R0)     ;WORD COUNT
016732  132760  000004  000074          BITB    #BIT02,$NCODE(R0) ;READ OP ?
016740  001004                          BNE     RANXIT            ;YES
                                ;GET A RANDOM PATTERN NUMBER

016742  004737  017122          RANPAT: JSR     PC,GETPAT         ;GET PATTERN CODE
016746  110560  000075                  MOVB    R5,$NPATC(R0)     ;MOVE PATTERN CODE TO CONTROL BLOCK
016752  012760  177777  000104  RANXIT: MOV     #-1,$NEXT(R0)    ;SET PARAMETERS SELECTED INDICATOR
016760  000207                          RTS     PC                ;RETURN

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

016762                          CHKADR:
016762  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
016764  010246                          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
```

```
016766  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016770  012701  000020                  MOV     #16.,R1         ;TOTAL 16 SETS OF BAD SECTOR ADDRESSES
016774  010002                          MOV     R0,R2           ;TABLE ADDRESS
016776  062702  000124                  ADD     #$BDSEC,R2      ;DBP ADDRESS + TABLE ADDRES OFFSET
017002  022712  177777          1$:     CMP     #-1,(R2)        ;EMPTY ENTRY ?
017006  001423                          BEQ     3$              ;BRANCH IF SO
017010  026012  000100                  CMP     $NCYL(R0),(R2)  ;ON THE SAME CYLINDER ?
017014  001013                          BNE     2$              ;BRANCH IF NOT
017016  126062  000011  000003          CMPB    $TRK(R0),3(R2)  ;ON THE SAME TRACK ?
017024  001007                          BNE     2$              ;BRANCH IF NOT
017026  126062  000010  000002          CMPB    $SEC(R0),2(R2)  ;ON THE SAME SECTOR ?
017034  001003                          BNE     2$              ;BRANCH IF NOT
017036  062705  000002                  ADD     #2,R5           ;OTHERWISE,TAKE ERROR EXIT
017042  000405                          BR      3$              ;EXIT
017044  005301                  2$:     DEC     R1              ;ALL 16 BAD SPOT CHECKED ?
017046  003403                          BLE     3$              ;BRANCH IF SO
017050  062702  000004                  ADD     #4,R2           ;ADJUST TO NEXT TABLE ENTRY
017054  000752                          BR      1$              ;LOOP BACK
017056                          3$:
017056  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
017060  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
017062  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
017064  000205                          RTS     R5              ;RETRUN

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

017066  012705  000002          RANWRT: MOV     #2,R5           ;SET WRITE DATA COMMAND
017072  005737  001474                  TST     FORMAT          ;ALLOW FORMAT OPERATION ?
017076  001406                          BEQ     1$              ;NO
017100  122760  000004  000024          CMPB    #4,$CODE(R0)    ;PREVIOUS A READ DATA COMMAND ?
017106  001002                          BNE     1$              ;NO
017110  012705  000003                  MOV     #3,R5           ;SET A WRITE HEAD AND DATA COMMAND
017114  110560  000074          1$:     MOVB    R5,$NCODE(R0)   ;SELECT THE WRITE COMMAND
017120  000207                          RTS     PC              ;EXIT

                                ;ROUTINE TO SELECT A PATTERN

017122  012705  000020          GETPAT: MOV     #20,R5          ;SELECT PATTERN
017126  005737  001510                  TST     PATTEN          ;ENABLE RANDOM PATTERN SELECTION ?
017132  001403                          BEQ     2$              ;YES
017134  013705  001510                  MOV     PATTEN,R5       ;USE INDEXED PATTERN
017140  000407                          BR      1$              ;NO
017142  004737  027304          2$:     JSR     PC,GETREM       ;GET CODE
017146  005705                          TST     R5              ;WAS PATTERN ZERO SELECTED ?
017150  001003                          BNE     1$              ;BR IF NOT ZERO
017152  004737  033350                  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
017156  000761                          BR      GETPAT          ;TRY AGAIN
017160  006305                  1$:     ASL     R5              ;MAKE CODE INTO TABLE INDEX
017162  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
```

```
017164  010546                          GETPAR: MOV     R5,-(SP)            ;SAVE R5
017166  116060  000234  000027                  MOVB    $RMCS1(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
017174  142760  177701  000027                  BICB    #^C76,$PREVO(R0)        ;STRIP GO AND IE BITS
017202  032760  000006  000074                  BIT     #6,$NCODE(R0)   ;SEE IF NEXT OPERATION IS READ OR WRITE
017210  001007                                  BNE     1$              ;BR IF EITHER
017212  016060  000012  000034                  MOV     $CYL(R0),$PREVA+2(R0)  ;SAVE STARTING CYLINDER
017220  016060  000010  000032                  MOV     $SEC(R0),$PREVA(R0)  ;SAVE STARTING SECTOR AND TRACK
017226  000410                                  BR      22$
017230  004737  022262          1$:             JSR     PC,READOR       ;GET THE DECREMENTED SECTOR AND TRACK ADDRESSES
017234  012660  000034                          MOV     (SP)+,$PREVA+2(R0)      ;CYLINDER ADDRESS
017240  112660  000033                          MOVB    (SP)+,$PREVA+1(R0)  ;TRACK ADDRESS
017244  112660  000032                          MOVB    (SP)+,$PREVA(R0)  ;SECTOR ADDRESS
                                        ;       MOV     $CYL(R0),$PREVA+2(R0)  ;CURRENT CYLINDER
017250  032777  000100  161676  22$:            BIT     #SW06,@SWR      ;SWITCH 6 SET ?
017256  001073                                  BNE     5$              ;BR IF SET
017260  116060  000074  000024                  MOVB    $NCODE(R0),$CODE(R0)  ;LOGICAL CODE FOR OPERATION
017266  116005  000074                          MOVB    $NCODE(R0),R5   ;LOAD R5 FOR USE AS TABLE INDEX
017272  116560  002122  000002                  MOVB    COMTBL(R5),$COMND(R0)  ;RM03 COMMAND CODE
017300  122760  000151  000002                  CMPB    #151,$COMND(R0) ;WRITE CHECK DATA COMMAND ?
017306  001013                                  BNE     3$              ;NO,DONOT CARE
017310  122760  000060  000027                  CMPB    #60,$PREVO(R0)  ;PREVIOUS A WRITE DATA COMMAND ?
017316  001420                                  BEQ     4$              ;YES,O K
017320  112760  000171  000002  2$:             MOVB    #171,$COMND(R0) ;CHANG TO READ DATA COMMAND
017326  112760  000004  000024                  MOVB    #4,$CODE(R0)    ;CODE NUMBER CHANGED TO READ DATA
017334  000411                                  BR      4$              ;EXIT
017336  122760  000153  000002  3$:             CMPB    #153,$COMND(R0) ;WRITE CHECK HEAD AND DATA COMMAND ?
017344  001005                                  BNE     4$              ;NO,THEN EXIT
017346  122760  000062  000027                  CMPB    #62,$PREVO(R0)  ;PREVIOUS A WRITE HEAD AND DATA COMMAND?
017354  001401                                  BEQ     4$              ;YES,EXIT
017356  000760                                  BR      2$              ;SET TO READ DATA COMMAND
017360                                  4$:
017360  116060  000075  000030                  MOVB    $NPATC(R0),$PATTC(R0)  ;PATTERN CODE
017366  016060  000076  000010                  MOV     $NSEC(R0),$SEC(R0)  ;TRACK AND SECTOR ADDRESSES
017374  016060  000100  000012                  MOV     $NCYL(R0),$CYL(R0)  ;CYLINDER ADDRESS
017402  016060  000102  000020                  MOV     $NWRDL(R0),$WRDL(R0)  ;BUFFER SIZE
017410  016060  000102  000004                  MOV     $NWRDL(R0),$WRDM(R0)  ;WORD COUNT FOR THE RH11
017416  005460  000004                          NEG     $WRDM(R0)       ;COMPLEMENT IT
017422  012760  000400  000022                  MOV     #256.,$SSEC(R0) ;INITIAL VALUE OF SECTOR SIZE
017430  032760  000001  000024                  BIT     #1,$CODE(R0)    ;HEADER OPERATION ?
017436  001403                                  BEQ     5$              ;BR IF NOT
017440  062760  000002  000022                  ADD     #2,$SSEC(R0)    ;ADD HEADER SIZE
017446  005060  000104          5$:             CLR     $NEXT(R0)       ;RESET 'PARAMETERS LOADED' INDICATOR
017452  012605                                  MOV     (SP)+,R5        ;RESTORE R5
017454  000207                                  RTS     PC              ;RETURN

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

017456  016111  000002          CMPRES: MOV     2(R1),(R1)      ;COMPRESS THE TABLE IN R1
017462  001403                          BEQ     1$              ;BR WHEN ZERO FOUND
017464  062701  000002                  ADD     #2,R1           ;INCREMENT R1
017470  000772                          BR      CMPRES          ;CONTINUE COMPRESSING TABLE
```

```
017472  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

017474  004737  033350          WRTPK:  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
017500  005760  000040                  TST     $OPERC+2(R0)    ;SEE IF FIRST OPERATION
017504  001007                          BNE     WRTPK1          ;BR IF UPPER WORD OF COUNTER NOT ZERO
017506  005760  000036                  TST     $OPERC(R0)      ;LOWER WORD ZERO ?
017512  001004                          BNE     WRTPK1          ;BR IF NOT 1ST OPERATION
017514  105760  000026                  TSTB    $PACK(R0)       ;SEE WHICH - 'R' OR 'W'
017520  100530                          BMI     WRTPK3          ;BR IF 'W'
017522  000515                          BR      WRTPK2          ;'R' OPERATION
017524  116060  000234  000027  WRTPK1: MOVB    $RMCSI(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
017532  004737  022262                  JSR     PC,READDR       ;GET THE DECREMENTED SECTOR AND TRACK ADDRESSES
017536  012660  000034                  MOV     (SP)+,$PREVA+2(R0)     ;CYLINDER ADDRESS
017542  112660  000033                  MOVB    (SP)+,$PREVA+1(R0)  ;TRACK ADDRESS
017546  112660  000032                  MOVB    (SP)+,$PREVA(R0)  ;SECTOR ADDRESS
017552  016060  000242  000010          MOV     $RMDA(R0),$SEC(R0)  ;NEW SECTOR & TRACK ADDRESS
017560  016060  000270  000012          MOV     $RMDC(R0),$CYL(R0)  ;NEW CYLINDER ADDRESS
017566  026060  000012  000106          CMP     $CYL(R0),MAXCYL(R0)  ;SEE IF AT END
017574  103436                          BLO     2$              ;BR IF LESS THAN 'MAXCYL'
017576  101004                          BHI     1$              ;BR IF GREATER THAN 'MAXCYL'
017600  126060  000011  000112          CMPB    $TRK(R0),MAXTRK(R0)  ;SEE IF AT MAX TRACK
017606  103431                          BLO     2$              ;BR IF NOT GREATER
017610  116060  000114  000011  1$:     MOVB    MINTRK(R0),$TRK(R0)  ;RESET TRACK ADDRESS
017616  116060  000120  000010          MOVB    MINSEC(R0),$SEC(R0)  ;RESET SECTOR ADDRESS
017624  016060  000110  000012          MOV     MINCYL(R0),$CYL(R0)  ;RESET CYLINDER ADDRESS
017632  112760  000004  000024          MOVB    #4,$CODE(R0)    ;SET CODE TO READ DATA
017640  122760  177776  000026          CMPB    #-2,$PACK(R0)   ;WT OPERATION IN PROCESSING
017646  001475                          BEQ     WRTPK5          ;YES
017650  004737  026776                  JSR     PC,EOP2         ;DROP THE DRIVE (NORMAL TERMINATION)
017654  032777  000020  161272          BIT     #SW04,@SWR      ;IS SWITCH 4 SET ?
017662  001003                          BNE     2$              ;BR IF SET
017664  005726                          TST     (SP)+           ;INCREMENT THE STACK POINTER
017666  000137  005352                  JMP     MAIN            ;RETURN DIRECTLY TO 'MAIN'
017672  013760  001462  000020  2$:     MOV     MAXDL,$WRDL(R0)  ;BUFFER SIZE IS MAXIMUM
017700  042760  000377  000020          BIC     #377,$WRDL(R0)  ;SECTOR BOUNDRY FOR WRITTING
017706  013760  001462  000004          MOV     MAXDL,$WRDM(R0)  ;WORD COUNT
017714  042760  000377  000004          BIC     #377,$WRDM(R0)  ;SECTOR BOUNDRY FOR WRITTING
017722  005760  000004                  TST     $WRDM(R0)       ;SIZE=0 ?
017726  003006                          BGT     3$              ;NO
017730  012760  000400  000004          MOV     #400,$WRDM(R0)  ;SET TO ONE SECTOR
017736  012760  000400  000020          MOV     #400,$WRDL(R0)  ;SET ONE SECTOR
017744  005460  000004          3$:     NEG     $WRDM(R0)       ;CHANGE WORD COUNT TO 2'S COMPLEMENT
017750  105760  000026                  TSTB    $PACK(R0)       ;READ OR WRITE ?
017754  100412                          BMI     WRTPK3          ;BR IF WRITE
017756  012760  000402  000022  WRTPK2: MOV     #258.,$SSEC(R0)  ;SECTOR SIZE FOR READ
017764  112760  000005  000024          MOVB    #5,$CODE(R0)    ;CODE FOR READ HEADER & DATA
017772  112760  000173  000002          MOVB    #RDHD,$COMND(R0)  ;DRIVE CODE FOR OPERATION
```

```
020000  000415                                  BR      WRTPK4          ;SET UP FOR EXIT
020002  012760  000400  000022  WRTPK3:  MOV     #256.,$SSEC(R0)  ;SECTOR SIZE
020010  112760  000002  000024          MOVB    #2,$CODE(R0)     ;CODE FOR WRTDAT
020016  112760  000161  000002          MOVB    #WRTDAT,$COMND(R0)  ;OP CODE
020024  004737  017122                  JSR     PC,GETPAT        ;GET PATTERN CODE
020030  110560  000030                  MOVB    R5,$PATTC(R0)    ;PATTERN CODE
020034  005060  000104  WRTPK4:  CLR     $NEXT(R0)        ;CLEAR 'PARAMETER SELECTED' INDICATOR
020040  000207                          RTS     PC               ;RETURN
020042  005037  001316  WRTPK5:  CLR     PACK             ;CLEAR WT FLAG
020046  105060  000026                  CLRB    $PACK(R0)        ;CLEAR WT FLAG
020052  005726                          TST     (SP)+            ;CLEAR STACK LEVEL
020054  000137  005352                  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

020060                          SPOTCK:
020060  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
020062  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
020064  012701  000124                  MOV     #$BDSEC,R1       ;INCREMENT FOR BAD SECTOR TABLE
020070  060001                          ADD     R0,R1            ;ADD THE BLOCK'S STARTING ADDRESS
020072  012702  000020                  MOV     #16.,R2          ;BAD SECTOR TABLE SIZE COUNT
020076  004737  022262          1$:     JSR     PC,READDR        ;DECREMENT THE SECTOR/TRACK ADDRESS
020102  021126                          CMP     (R1),(SP)+       ;ON THE SAME CYLINDER ?
020104  001007                          BNE     2$               ;BRANCH IF NOT
020106  122661  000003                  CMPB    (SP)+,3(R1)      ;COMPARE THE TRACK ADDRESS
020112  001005                          BNE     3$               ;BR IF IT IS NOT EQUAL
020114  122661  000002                  CMPB    (SP)+,2(R1)      ;COMPARE THE SECTOR ADDRESS
020120  001003                          BNE     4$               ;BR IF NOT EQUAL
020122  000411                          BR      5$               ;CHECK 'NOTPRT'
020124  005726                  2$:     TST     (SP)+            ;CLEAR OFF THE STACK
020126  005726                  3$:     TST     (SP)+            ;INCREMENT THE STACK POINTER
020130  062701  000004          4$:     ADD     #4,R1            ;GO TO THE NEXT LOCATION IN THE TABLE
020134  005711                          TST     (R1)             ;PAST THE TABLE ENTRIES ?
020136  100411                          BMI     6$               ;BR IF PAST
020140  005302                          DEC     R2               ;DECREMENT THE MAXIMUM ENTRY COUNT
020142  001355                          BNE     1$               ;BR IF MORE TO CHECK
020144  000406                          BR      6$               ;END, EXIT
020146  005737  001504          5$:     TST     NOTPRT           ;PRINT THE ERROR ANYWAY ?
020152  001006                          BNE     7$               ;BR IF NOT
020154  012737  177777  001362          MOV     #-1,BADSEC       ;SET THE INDICATOR FOR THE IDENTIFICATION LINE
020162  062766  000002  000004  6$:     ADD     #2,4(SP)         ;INCREMENT THE RETURN
020170                          7$:
020170  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
020172  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
020174  000207                          RTS     PC               ;RETURN

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

                                .SBTTL  ERROR MESSAGE GENERATION ROUTINES
```

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

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

020176  032777  002000  160750  LINE1:  BIT     #SW10,@SWR      ;SWITCH 10 SET ?
020204  001402                          BEQ     1$              ;BR IF NOT
020206  104401  001174                  TYPE    ,$BELL          ;RING THE BELL
020212  032777  020000  160734  1$:     BIT     #SW13,@SWR      ;INHIBIT TYPEOUT ?
020220  001403                          BEQ     2$              ;BR IF NOT
020222  104414  001201                  DISPLY  ,$CRLF          ;CR-LF
020226  000404                          BR      3$              ;EXIT
020230  004737  023450          2$:     JSR     PC,$TIME        ;TYPE THE TIME
020234  104414  052717                  DISPLY  ,LINSPO         ;SPACES
020240  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'

020242                          LINE2:
020242  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020244  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
020246  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020250  104414  001201                  DISPLY  ,$CRLF          ;CR-LF
020254  005037  020402                  CLR     4$              ;CLEAR MESSAGE ADDRESS STORAGE
020260  005004                          CLR     R4              ;WORKING REGISTER
020262  012737  051022  020402          MOV     #LIN2C,4$       ;ADDRESS OF 'PRESENT ORDER = ' MSG
020270  116004  000234                  MOVB    $RMCS1(R0),R4   ;GET THE OPCODE
020274  042704  177701                  BIC     #^C76,R4        ;SAVE ONLY SIGNIFICANT BITS
020300  004737  020336                  JSR     PC,1$           ;TYPE THE FIRST MNEMONIC
020304  005737  020406                  TST     5$              ;SEE IF MNEMONIC ENTRY FOUND
020310  001440                          BEQ     LINE2A          ;BR IF NOT
020312  012737  051043  020402          MOV     #LIN2P,4$       ;ADDRESS OF 'PREVIOUS ORDER = ' MSG
020320  116004  000027                  MOVB    $PREVO(R0),R4   ;PREVIOUS OPERATION CODE
020324  042704  177701                  BIC     #^C76,R4        ;SAVE ONLY SIGNIFICANT BITS
020330  004737  020336                  JSR     PC,1$           ;TYPE THE PREVIOUS MNEMONIC
020334  000426                          BR      LINE2A          ;CONTINUE
020336  005005                  1$:     CLR     R5              ;CLEAR THE TABLE INDEX
020340  126504  002130          2$:     CMPB    OPTBL(R5),R4    ;LOOK FOR THE OPCODE
020344  001405                          BEQ     3$              ;BR WHEN OPCODE COUNT EQUALS OPCODE
020346  105765  002130                  TSTB    OPTBL(R5)       ;LOOK FOR END OF TABLE
020352  100402                          BMI     3$              ;BR IF END
020354  005205                          INC     R5              ;INCREMENT THE POINTER
020356  000770                          BR      2$              ;CONTINUE - NOT END OF TABLE
020360  006305                  3$:     ASL     R5              ;SHIFT INDEX
020362  006305                          ASL     R5              ;SHIFT THE INDEX
020364  006305                          ASL     R5              ;SHIFT THE INDEX
020366  012737  002152  020406          MOV     #MNTBL,5$       ;ADDRESS OF ASCII TEXT TABLE
020374  060537  020406                  ADD     R5,5$           ;ADD THE INDEX
```

```
020400  104414                          DISPLY                  ;TYPE IT
020402  000000                  4$:     .WORD   0               ;ADDRESS OF 'PRESENT' OR 'PREVIOUS' MESSAGE
020404  104414                          DISPLY                  ;TYPE THE OPERATION MNEMONIC
020406  000000                  5$:     .WORD   0               ;ADDRESS OF MESSAGE
020410  000207                          RTS     PC              ;RETURN TO MAIN ROUTINE
020412  005737  001362          LINE2A: TST     BADSEC          ;PRINT THE BAD SECTOR LINE ?
020416  001404                          BEQ     LINE2B          ;BR IF NOT
020420  104414  001201                  DISPLY  ,$CRLF          ;CR-LF
020424  104414  051067                  DISPLY  ,LIN2S          ;ERROR ADDRESS DEFINED AS BAD AREA
020430  104414  001201          LINE2B: DISPLY  ,$CRLF          ;CR-LF
020434  104414  050416                  DISPLY  ,DH14           ;STANDARD RM03 REGISTER HEADER
020440  104414  052717                  DISPLY  ,LINSPO         ;TYPE A SPACE
020444  013746  001344                  MOV     UNIT,-(SP)      ;PUT THE DRIVE NUMBER ON THE STACK
020450  004737  022242                  JSR     PC,LINDEC       ;TYPE DRIVE NUMBER
020454  104414  052716                  DISPLY  ,LINSP          ;SPACES
020460  012705  050742                  MOV     #DT14,R5        ;REGISTER INDEXES
020464  004737  020614                  JSR     PC,3$           ;PRINT THE REGISTERS
020470  032777  000040  160456          BIT     #SWO5,@SWR      ;PRINT THE OPTIONAL REGISTERS ?
020476  001014                          BNE     1$              ;BR IF NOT
020500  104414  050521                  DISPLY  ,DH15
020504  012705  050764                  MOV     #DT15,R5        ;SECOND DATA LINE
020510  004737  020614                  JSR     PC,3$           ;PRINT THEM
020514  104414  050620                  DISPLY  ,DH16
020520  012705  051006                  MOV     #DT16,R5        ;THIRD DATA LINE
020524  004737  020614                  JSR     PC,3$           ;PRINT THE REGISTERS
020530  032760  000100  000016  1$:     BIT     #BIT6,$STATUS(R0)  ;DATA ERROR ?
020536  001422                          BEQ     2$              ;BR IF NOT
020540  016046  000020                  MOV     $WRDL(R0),-(SP) ;TRANSFER SIZE
020544  066016  000236                  ADD     $RMWC(R0),(SP)  ;ADD REMAINING WORD COUNT
020550  006316                          ASL     (SP)            ;CONVERT TO AN BYTE INCREMENT
020552  066016  000006                  ADD     $BUF(R0),(SP)   ;BUFFER STARTING ADDRESS
020556  022660  000240                  CMP     (SP)+,$RMBA(R0) ;CORRECT BUFFER ADDRESS ?
020562  001410                          BEQ     2$              ;BR IF YES
020564  104414  050043                  DISPLY  ,EM46           ;'BUS ADDRESS AND WORD COUNT ARE NOT CONSISTENT'
020570  104414  001201                  DISPLY  ,$CRLF          ;CR-LF
020574  004737  020706                  JSR     PC,LINE3D       ;PRINT LINE 3D OF ERROR MESSAGE
020600  004737  021320                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
020604                          2$:
020604  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
020606  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
020610  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
020612  000207                          RTS     PC              ;RETURN TO ERROR PROCESSING ROUTINE
020614  012546                  3$:     MOV     (R5)+,-(SP)     ;PUT THE REGISTER INDEX ON THE STACK
020616  060016                          ADD     R0,(SP)         ;ADD DRIVE'S TABLE ADDRESS
020620  017646  000000                  MOV     @(SP),-(SP)     ;VALUE
020624  004737  022210                  JSR     PC,LINOCT       ;TYPE IT
020630  005726                          TST     (SP)+           ;CORRECT THE STACK POINTER
020632  104414  052716                  DISPLY  ,LINSP          ;PRINT 2 SPACES
020636  005715                          TST     (R5)            ;AT END OF LINE ?
020640  001365                          BNE     3$              ;BR IF NOT
020642  104414  001201          4$:     DISPLY  ,$CRLF          ;CR-LF
020646  000207                          RTS     PC              ;RETURN

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

```
020650  104414  051123          LINE3:  DISPLY  ,LINM3          ;LINE 3 ENTRANCE
020654  000517                          BR      LIN3.1          ;FINISH PRINTOUT

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

020656  104414  051141          LINE3A: DISPLY  ,LINM3          ;LINE 3A ENTRANCE
020662  000514                          BR      LIN3.1          ;FINISH ERROR LINE

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

020664  004737  021222          LINE3B: JSR     PC,LIN3.3       ;LINE 3B ENTRANCE
020670  104414  001201                  DISPLY  ,$CRLF
020674  000207                          RTS     PC

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

020676  004737  021222          LINE3C: JSR     PC,LIN3.3       ;LINE 3C ENTRANCE
020702  000137  021254                  JMP     LIN3.4          ;FINISH MESSAGE

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

020706  032777  000040  160240  LINE3D: BIT     #SW05,@SWR      ;SWITCH 5 SET ?
020714  001416                          BEQ     1$              ;BR IF IT IS
020716  104414  051312                  DISPLY  ,LINB3          ;'RMBA = '
020722  016046  000240                  MOV     $RMBA(R0),-(SP) ;BUFFER ADDR REG CONTENTS
020726  004737  022210                  JSR     PC,LINOCT       ;CONVERT TO OCTAL AND TYPE IT
020732  104414  051322                  DISPLY  ,LINW3          ;'  RMWC = '
020736  016046  000236                  MOV     $RMWC(R0),-(SP) ;WORD COUNT REGISTER CONTENTS
020742  004737  022210                  JSR     PC,LINOCT       ;CONVERT TO OCTAL AND TYPE IT
020746  104414  001201                  DISPLY  ,$CRLF
020752  000207                  1$:     RTS     PC

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

020754  104414  051206          LINE3E: DISPLY  ,LINS3          ;'START CYL = '
020760  016046  000012                  MOV     $CYL(R0),-(SP)  ;MOVE CYL TO STACK
020764  004737  022242                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
020770  104414  052716                  DISPLY  ,LINSP          ;SPACES
020774  104414  051334                  DISPLY  ,LINST3         ;'START TRK = '
021000  005046                          CLR     -(SP)           ;CLEAR STACK
021002  116016  000011                  MOVB    $TRK(R0),(SP)   ;TRACK TO STACK
021006  004737  022242                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021012  104414  052716                  DISPLY  ,LINSP          ;SPACES
021016  104414  051351                  DISPLY  ,LINSS3         ;'START SEC = '
021022  005046                          CLR     -(SP)           ;CLEAR STACK
021024  116016  000010                  MOVB    $SEC(R0),(SP)   ;SECTOR ADDR TO STACK
021030  004737  022242                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021034  104414  001201                  DISPLY  ,$CRLF
021040  000207                          RTS     PC

                                ;PRINT LINE 3F OF ERROR MESSAGE
```

```
                                        ;'RMDA = XXXXXX   RMCA = XXXXXX'

021042  032777  000040  160104  LINE3F: BIT     #SW5,@SWR        ;SWITCH 5 SET ?
021050  001420                          BEQ     1$               ;BR IF NOT
021052  104414  051302                  DISPLY  ,LINDA3          ;'RMDA = '
021056  016046  000242                  MOV     $RMDA(R0),-(SP)  ;PUT SECTOR/TRACK ADDRESS ON THE STACK
021062  004737  022210                  JSR     PC,LINOCT        ;TYPE IT
021066  104414  052716                  DISPLY  ,LINSP           ;SPACES
021072  104414  051271                  DISPLY  ,LINCA3          ;' RMDC = '
021076  016046  000270                  MOV     $RMDC(R0),-(SP)  ;PUT DESIRED CYLINDER ADDRESS ON THE STACK
021102  004737  022210                  JSR     PC,LINOCT        ;TYPE IT
021106  104414  001201                  DISPLY  ,$CRLF
021112  000207                  1$:     RTS     PC

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

                                        ;LIN3.1:        MOV     $RMDC(R0),-(SP) ;PUT CYLINDER ADDR ON STACK
021114  004737  022262          LIN3.1: JSR     PC,READDR        ;DECREMENT TRACK AND SECTOR ADDRESS
021120  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021124  104414  051136                  DISPLY  ,T               ;PRINT ' T'
                                        ;       JSR     PC,READDR        ;DECREMENT TRACK AND SECTOR ADDRESSES
021130  004737  022242                  JSR     PC,LINDEC        ;TYPE TRACK IN DECIMAL
021134  104414  051162                  DISPLY  ,S               ;PRINT ' S'
021140  004737  022242                  JSR     PC,LINDEC        ;TYPE SECTOR ADDRESS
021144  104414  051165                  DISPLY  ,LINP3           ;PRINT 'PREV ADDR'
021150  016046  000034                  MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
021154  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021160  104414  051136                  DISPLY  ,T               ;PRINT ' T'
021164  005046                          CLR     -(SP)            ;MAKE ROOM ON THE STACK
021166  116016  000033                  MOVB    $PREVA+1(R0),(SP)  ;PREVIOUS TRACK ADDRESS
021172  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021176  104414  051162                  DISPLY  ,S               ;PRINT ' S'
021202  005046                          CLR     -(SP)            ;MAKE ROOM ON THE STACK
021204  116016  000032                  MOVB    $PREVA(R0),(SP)  ;PREVIOUS SECTOR DORESS
021210  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021214  104414  001201                  DISPLY  ,$CRLF
021220  000207                          RTS     PC

                                        ;'START CYL = CCC   END CYL = CCC'

021222  104414  051206          LIN3.3: DISPLY  ,LINS3           ;LINE '3B & 3C' ENTRANCE
021226  016046  000034                  MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
021232  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021236  104414  051223                  DISPLY  ,LINEN3          ;PRINT 'END CYL'
021242  016046  000270                  MOV     $RMDC(R0),-(SP)  ;PRESENT CYLINDER
021246  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021252  000207                          RTS     PC

                                        ;'ACTUAL CYL = CCC   TRK = TT'

021254  104414  051240          LIN3.4: DISPLY  ,LINA3           ;PRINT 'ACTUAL'
021260  013746  056150                  MOV     CYLDER,-(SP)     ;ACTUAL CYLINDER
021264  042716  010000                  BIC     #BIT12,(SP)      ;CLEAR THE FORMAT BIT
021270  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021274  104414  051260                  DISPLY  ,LINT3           ;PRINT TRACK
021300  005046                          CLR     -(SP)            ;CLEAR STACK WORD
```

```
021302  116016  000243                  MOVB    $RMDA+1(R0),(SP) ;PUT TRACK ON STACK
021306  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021312  104414  001201                  DISPLY  ,$CRLF
021316  000207                          RTS     PC

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

021320  032760  000100  000016  LINE4:  BIT     #BIT06,$STATUS(R0)  ;DATA ERROR ?
021326  001427                          BEQ     1$               ;BR IF NOT
021330  104414  051366                  DISPLY  ,LINM4           ;'PRINT BUFFER'
021334  016046  000006                  MOV     $BUF(R0),-(SP)   ;BUFFER ADDR ON STACK
021340  004737  022210                  JSR     PC,LINOCT        ;CONVERT TO OCTAL & PRINT
021344  104414  051405                  DISPLY  ,LINS4           ;PRINT 'SIZE'
021350  016046  000020                  MOV     $WRDL(R0),-(SP)  ;BUFFER SIZE
021354  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021360  104414  051417                  DISPLY  ,LINX4           ;'ACTUAL NMBR WRDS XFRD = '
021364  016046  000240                  MOV     $RMBA(R0),-(SP)  ;VALUE IN BUFFER ADDR REGISTER
021370  166016  000006                  SUB     $BUF(R0),(SP)    ;SUBTRACT STARTING ADDRESS
021374  006216                          ASR     (SP)             ;CONVERT INTO A WORD COUNT
021376  004737  022242                  JSR     PC,LINDEC        ;TYPE IT IN DECIMAL
021402  104414  001201                  DISPLY  ,$CRLF           ;CR-LF
021406  000207                  1$:     RTS     PC               ;RETURN

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

021410  104414  051452          LINE5:  DISPLY  ,LINDS           ;PRINT 'GOOD DATA'
021414  162760  000002  000240          SUB     #2,$RMBA(R0)     ;BACK THE ADDRESS UP
021422  017046  000240                  MOV     @$RMBA(R0),-(SP) ;'GOOD' DATA - AT THE BUFFER LOCATION
021426  004737  022210                  JSR     PC,LINOCT        ;TYPE IT
021432  104414  051467                  DISPLY  ,LINBS           ;PRINT 'BAD DATA'
021436  016046  000256                  MOV     $RMDB(R0),-(SP)  ;BAD DATA FROM BUFFER
021442  004737  022210                  JSR     PC,LINOCT        ;TYPE IT
021446  016046  000236                  MOV     $RMWC(R0),-(SP)  ;WORD LENGTH ON STACK
021452  066016  000020                  ADD     $WRDL(R0),(SP)   ;MAKE INTO A POSITIVE NUMBER
021456  005046                          CLR     -(SP)            ;UPPER DIVIDEND TO ZERO
021460  016046  000022                  MOV     $SSEC(R0),-(SP)  ;SECTOR SIZE ON THE STACK
021464  004737  027332                  JSR     PC,LINKDV        ;DIVIDE WORDS XFERED BY SECTOR SIZE
021470  012616                          MOV     (SP)+,(SP)       ;MOVE REMAINDER UP THE STACK
021472  104414  051505                  DISPLY  ,LINPS           ;PRINT 'SECT POS'
021476  004737  022242                  JSR     PC,LINDEC        ;TYPE THE POSITION
021502  104414  001201                  DISPLY  ,$CRLF
021506  000207                          RTS     PC

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

021510                          LINE5A:
021510  104414  051523          2$:     DISPLY  ,LINS5           ;'HEADER CONTENTS OF ERROR SECTOR'
021514  013746  056150                  MOV     CYLDER,-(SP)     ;HEADER POSITION
021520  004737  022210                  JSR     PC,LINOCT        ;TYPE IT
021524  104414  052716                  DISPLY  ,LINSP           ;SPACES
021530  013746  056152                  MOV     CYLDER+2,-(SP)   ;HEADER POSITION +2
021534  004737  022210                  JSR     PC,LINOCT        ;TYPE IT
021540  104414  052716                  DISPLY  ,LINSP           ;SPACES
```

```
021544  104414  052721                  DISPLY  ,LINX5          ;APPENDING INFO 1/23/77
021550  104414  001201          3$:     DISPLY  ,$CRLF
021554  000207                          RTS     PC

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

021556  104414  051557          LINE5B: DISPLY  ,LINEP5         ;'RMEC1 = '
021562  016046  000300                  MOV     $RMEC1(R0),-(SP) ;PUT REGISTER CONTENTS ON THE STACK
021566  004737  022210                  JSR     PC,LINOCT       ;TYPE IT
021572  104414  052716                  DISPLY  ,LINSP          ;SPACES
021576  104414  051570                  DISPLY  ,LINE05         ;' RMEC2 = '
021602  016046  000302                  MOV     $RMEC2(R0),-(SP) ;PUT REGISTER CONTENTS ON THE STACK
021606  004737  022210                  JSR     PC,LINOCT       ;TYPE IT
021612  104414  001201                  DISPLY  ,$CRLF
021616  000207                          RTS     PC              ;RETURN

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

021620  104414  051602          LINE6:  DISPLY  ,LINB6          ;ECC CORRECTABLE
021624  104414  001201                  DISPLY  ,$CRLF
021630  000207                          RTS     PC

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

021632  104414  051635          LINE6A: DISPLY  ,LINC6          ;PRINT 'READ CORRECTLY AT OFFSET N'
021636  000411                          BR      LIN6.1          ;TYPE THE REST OF THE LINE

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

021640  104414  051602          LINE6B: DISPLY  ,LINB6          ;PRINT 'SECTOR IS ECC CORRECTABLE '
021644  000406                          BR      LIN6.1

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

021646  104414  051664          LINE6C: DISPLY  ,LING6          ;'CORRECTED ON NTH RETRY'
021652  000414                          BR      LIN6.2          ;TYPE THE REST OF THE LINE

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

021654  104414  051713          LINE6D: DISPLY  ,LINU06         ;'UNCORRECTABLE AFTER N RETRIES'
021660  000411                          BR      LIN6.2          ;FINISH

                                ;TYPE THE OFFSET VALUE IN MICRO-INCHES

021662  006301                  LIN6.1: ASL     R1              ;DOUBLE THE OFFSET TABLE INDEX
021664  016137  002470  021674          MOV     OFMTBL(R1),1$   ;ADDRESS OF OFFSET POSITION MESSAGE
021672  104414                          DISPLY
021674  000000                  1$:     .WORD   0               ;OFFSET VALUE
021676  104414  001201                  DISPLY  ,$CRLF
021702  000207                          RTS     PC
```

```
                                        ;RETRY COUNT TYPEOUT

021704  005046                          LIN6.2: CLR     -(SP)           ;CLEAR STACK
021706  113716  001351                          MOVB    RETRY+1,(SP)    ;RETRY COUNT
021712  004737  022242                          JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021716  104414  051702                          DISPLY  ,LIN6           ;'RETRY'
021722  104414  001201                          DISPLY  ,$CRLF
021726  000207                                  RTS     PC

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

021730  104414  051766                  LINE7:  DISPLY  ,LIN7O          ;PRINT ORDER COUNT
021734  012746  000036                          MOV     #$OPERC,-(SP)   ;TO STACK
021740  060016                                  ADD     R0,(SP)         ;ADD THE BASE ADDRESS
021742  004737  033546                          JSR     PC,$DB2D        ;CONVERT IT
021746  004737  027646                          JSR     PC,$SUPRS       ;PRINT IT
021752  104414  052040                          DISPLY  ,LIN7T          ;TOTAL ERRORS
021756  016046  000056                          MOV     $TOTAL(R0),-(SP)        ;TO STACK
021762  004737  022242                          JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021766  104414  052052                          DISPLY  ,LIN7X          ;PRINT 'WRDS XFR'
021772  012746  000046                          MOV     #$TRANS,-(SP)   ;ADDRESS OF LOW WORD ON STACK
021776  060016                                  ADD     R0,(SP)
022000  004737  033546                          JSR     PC,$DB2D        ;CONVERT
022004  004737  027646                          JSR     PC,$SUPRS       ;PRINT
022010  104414  052066                          DISPLY  ,LIN7R          ;'BITS READ'
022014  012746  000052                          MOV     #$READ,-(SP)    ;LOW WORD ADDRESS
022020  060016                                  ADD     R0,(SP)
022022  004737  033546                          JSR     PC,$DB2D        ;CONVERT
022026  004737  027646                          JSR     PC,$SUPRS       ;PRINT
022032  104414  001201                          DISPLY  ,$CRLF
022036  104414  001201                          DISPLY  ,$CRLF
022042  032777  100000  157104                  BIT     #SW15,@SWR      ;SEE IF 'HALT ON ERROR' - SWITCH 15
022050  001401                                  BEQ     1$              ;BR IF NOT
022052  000000                                  HALT                    ;SWITCH 15 HALT
022054  000207                          1$:     RTS     PC

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

022056  104414  051766                  LINE7A: DISPLY  ,LIN7O          ;'ORDERS = '
022062  012746  000036                          MOV     #$OPERC,-(SP)   ;ORDER COUNT INCREMENT
022066  060016                                  ADD     R0,(SP)         ;ADD BASE ADDRESS
022070  004737  033546                          JSR     PC,$DB2D        ;CONVERT THE COUNT
022074  004737  027646                          JSR     PC,$SUPRS       ;PRINT IT
022100  104414  051776                          DISPLY  ,LIN7P          ;'TOTAL SEEKS = '
022104  012746  000042                          MOV     #$POSIT,-(SP)   ;TOTAL SEEKS
022110  060016                                  ADD     R0,(SP)         ;DEVICE TABLE ADDRESS
022112  004737  033546                          JSR     PC,$DB2D        ;CONVERT THE SEEK COUNT
022116  004737  027646                          JSR     PC,$SUPRS       ;PRINT IT
022122  104414  051740                          DISPLY  ,LIN7M          ;'  TOTAL MISPOS ERR = '
022126  016046  000066                          MOV     $MISPO(R0),-(SP)  ;TOTAL ERRORS
022132  004737  022242                          JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
022136  104414  052016                          DISPLY  ,LIN7S          ;'  TOTAL SKI,OCYL ERR = '
022142  016046  000064                          MOV     $SKI(R0),-(SP)  ;CONVERT & PRINT IT
```

```
022146  004737  022242                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
022152  104414  001201                  DISPLY  ,$CRLF
022156  104414  001201                  DISPLY  ,$CRLF
022162  032777  100000  156764          BIT     #SW15,@SWR      ;SEE IF HALT ON ERROR - SWITCH 15 SET
022170  001401                          BEQ     1$              ;BR IF NOT
022172  000000                          HALT                    ;SWITCH 15 HALT
022174  000207                  1$:     RTS     PC

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

022176  104414  052103          LINE8:  DISPLY  ,LIN8M
022202  004737  020242                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
022206  000207                          RTS     PC

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

022210  016646  000002          LINOCT: MOV     2(SP),-(SP)     ;PUT NUMBER IN PROPER LOCATION ON STACK
022214  004737  030276                  JSR     PC,$SB2O        ;CONVERT THE NUMBER TO OCTAL
022220  012637  022234                  MOV     (SP)+,1$        ;GET THE ADDRESS OF THE ASCII STRING
022224  062737  000005  022234          ADD     #5.,1$          ;ADDRESS THE LAST 6 ASCII DIGITS
022232  104414                          DISPLY                  ;TYPE IT
022234  000000                  1$:     .WORD   0               ;ADDRESS
022236  012616                          MOV     (SP)+,(SP)      ;CORRECT THE STACK
022240  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

022242  016646  000002          LINDEC: MOV     2(SP),-(SP)     ;SET UP STACK FOR CONVERT
022246  004737  030246                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
022252  004737  027646                  JSR     PC,$SUPRS       ;TYPE IT (WITH LEADING ZEROS SUPRESSED)
022256  012616                          MOV     (SP)+,(SP)      ;RESTORE STACK POINTER
022260  000207                          RTS     PC

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

                                .SBTTL  GENERAL SUPPORT SUBROUTINES

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

                                ;DECREMENT THE SECTOR-TRACK ADDRESS
                                ;CALL:
                                ;       MOV     #DPB,R0         ;DPB ADDRESS
                                ;       JSR     PC,READDR
                                ;       RETURN
                                ;       (SP) CONTAINS THE CYLINDER ADDRESS
                                ;       2(SP) CONTAINS THE TRACK ADDRESS
```

```
                                        ;       4(SP) CONTAINS THE SECTOR ADDRESS

022262  162706  000006          READDR: SUB     #6,SP           ;DECREMENT THE STACK POINTER
022266  016616  000006                  MOV     6(SP),(SP)      ;MOVE THE RETURN ADDR DOWN THE STACK
022272  005066  000006                  CLR     6(SP)           ;CLEAR STACK FOR SECTOR
022276  005066  000004                  CLR     4(SP)           ;CLEAR STACK FOR TRACK
022302  005066  000002                  CLR     2(SP)           ;CLEAR STACK FOR CYLINDER
022306  116066  000242  000006          MOVB    $RMDA(R0),6(SP)  ; SECTOR ON STACK
022314  116066  000243  000004          MOVB    $RMDA+1(R0),4(SP)  ;TRACK ADDRESS
022322  016066  000270  000002          MOV     $RMDC(R0),2(SP) ;CYLINDER ADDRESS
022330  005766  000006                  TST     6(SP)           ;SECTOR 0 ?
022334  001403                          BEQ     1$              ;BRANCH IF SO
022336  105366  000006                  DECB    6(SP)           ;DECREMENT ONE SECTOR
022342  000421                          BR      3$              ;BRANCH TO EXIT
022344  005766  000004          1$:     TST     4(SP)           ;ALSO ON TRACK 0 ?
022350  001406                          BEQ     2$              ;BRANCH IF SO
022352  112766  000037  000006          MOVB    #31.,6(SP)      ;LAST SECTOR
022360  105366  000004                  DECB    4(SP)           ;DECREMENT ONE TRACK
022364  000410                          BR      3$              ;EXIT
022366  112766  000037  000006  2$:     MOVB    #31.,6(SP)      ;LAST SECTOR
022374  112766  000004  000004          MOVB    #4,4(SP)        ;LAST TRACK
022402  005366  000002                  DEC     2(SP)           ;DECREMENT ONE CYLINDER COUNT
022406  000240                  3$:     NOP                     ;DONE
022410  000207                          RTS     PC              ;RETURN

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

022412  012737  177777  001310  CKCLK:  MOV     #-1,CLKFLG      ;CLEAR CLOCK AVAILABILITY FLAG
022420  012737  177777  001306          MOV     #-1,PCLOCK      ;CLEAR KW11-P CLOCK AVAILABILITY FLAG
022426  012737  022506  000004          MOV     #CKCLK1,ERRVEC  ;SET UP VECTOR FOR CLOCK CHECK
022434  005037  000006                  CLR     @#ERRVEC+2        ;NEW PSW
022440  005777  156630                  TST     @$LKCSR         ;CHECK FOR KW11-P
022444  005037  001310                  CLR     CLKFLG          ;SET CLOCK AVAILABILITY FLAG
022450  005037  001306                  CLR     PCLOCK          ;SET KW11-P CLOCK FLAG
022454  013701  001300                  MOV     $LPVEC,R1       ;KW11-P VECTOR ADDRESS
022460  012721  023546                  MOV     #CLOCK,(R1)+    ;SET UP KW11-P VECTOR
022464  012711  000300                  MOV     #300,(R1)       ;PSW - PRI 6
022470  012777  174575  156600          MOV     #-1667.,@$LKCSB ;LOAD COUNTER BUFFER WITH 16.67
022476  012777  000131  156570          MOV     #131,@$LKCSR    ;SET CLOCK - CNT UP, 10US, CONT INT
022504  000437                          BR      CKCLK3
022506  062706  000004          CKCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
022512  012737  022554  000004          MOV     #CKCLK2,@#ERRVEC  ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
022520  005777  156556                  TST     @$LKS           ;LOOK FOR KW11-L
022524  005037  001310                  CLR     CLKFLG          ;SET CLOCK FLAG
022530  013701  001304                  MOV     $LLVEC,R1       ;KW11-L VECTOR ADDRESS
022534  012721  023546                  MOV     #CLOCK,(R1)+    ;SET UP KW11-L VECTOR
022540  012711  000300                  MOV     #300,(R1)       ;PSW - PRI 6
022544  012777  000100  156530          MOV     #100,@$LKS      ;SET KW11-L INTERRUPT
022552  000414                          BR      CKCLK3
022554  062706  000004          CKCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
022560  104401  053512                  TYPE    ,NEDCLK         ;'P OR L CLOCK MUST BE ON SYSTEM'
022564  005737  000042                  TST     42              ;UNDER MONITOR CONTROL ?
022570  001402                          BEQ     1$              ;BR IF NOT
022572  000137  027250                  JMP     $GET42          ;ABORT PROGRAM
022576  000000                  1$:     HALT                    ;HALT
022600  000137  003622                  JMP     START           ;TRY AGAIN
```

```
022604  012737  000006  000004  CKCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
022612  000207                          RTS     PC

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

022614  010046                  STATPR: MOV     R0,-(SP)        ;SAVE R0
022616  010446                          MOV     R4,-(SP)        ;SAVE R4
022620  005737  001544                  TST     ASNLST          ;ANY DRIVES ASSIGNED ?
022624  001423                          BEQ     3$              ;BR IF NOT
022626  004737  022730                  JSR     PC,SHDTYP       ;TYPE THE HEADING
022632  005004                          CLR     R4              ;CLEAR THE DRIVE INDEX
022634  006304                  1$:     ASL     R4              ;CHANGE TO INDEX WORDS
022636  016400  002102                  MOV     BLKADR(R4),R0   ;GET THE DRIVE'S BLOCK ADDRESS
022642  006204                          ASR     R4              ;RESTORE R4
022644  136437  034304  001544          BITB    ATABIT(R4),ASNLST  ;IS THIS DRIVE ASSIGNED ?
022652  001404                          BEQ     2$              ;BR IF NOT
022654  004737  022752                  JSR     PC,SDETAL       ;TYPE THE PERFORMANCE SUMMARY
022660  104401  001201                  TYPE    ,$CRLF          ;CR-LF
022664  005204                  2$:     INC     R4              ;INCREMENT THE INDEX
022666  020427  000010                  CMP     R4,#8.          ;FINISHED ?
022672  001360                          BNE     1$              ;BR IF NOT
022674  012604                  3$:     MOV     (SP)+,R4        ;RESTORE R4
022676  012600                          MOV     (SP)+,R0        ;RESTORE R0
022700  000207                          RTS     PC              ;RETURN

                                ;ROUTINE TO TYPE STATISTICS FOR AN INDIVIDUAL DRIVE
                                ;CALL:
                                ;       MOV     #DPB,R0         ;DPB ADDRESS
                                ;       JSR     PC,TYPEST
                                ;       RETURN

022702  010046                  TYPEST: MOV     R0,-(SP)        ;SAVE R0
022704  010446                          MOV     R4,-(SP)        ;SAVE R4
022706  004737  022730                  JSR     PC,SHDTYP       ;TYPE THE HEADING
022712  005004                          CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
022714  111004                          MOVB    (R0),R4         ;DRIVE NUMBER
022716  004737  022752                  JSR     PC,SDETAL       ;TYPE THE STATISTICS
022722  012604                          MOV     (SP)+,R4        ;RESTORE R4
022724  012600                          MOV     (SP)+,R0        ;RESTORE R0
022726  000207                          RTS     PC              ;RETURN

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

022730  004737  023450          SHDTYP: JSR     PC,$TIME        ;TYPE THE TIME OF DAY
022734  004537  027706                  JSR     R5,TYPRI4       ;TYPE AT PRIORITY 4
022740  001201                          $CRLF                   ;CR-LF
022742  004537  027706                  JSR     R5,TYPRI4       ;TYPE THE HEADER
022746  053162                          STATHD                  ;HEADER
022750  000207                          RTS     PC              ;RETURN
```

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

022752  010246                  SDETAL: MOV     R2,-(SP)        ;SAVE R2
022754  010002                          MOV     R0,R2           ;DPB ADDRESS
022756  062702  000036                  ADD     #$OPERC,R2      ;FIRST STATISTICAL FIELD
022762  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
022764  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
022766     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
022767     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
022770  104401  052716                  TYPE    ,LINSP          ;SPACES
022774  016046  000070                  MOV     $PASSC(R0),-(SP)  ;PUT THE PASS COUNT ON THE STACK
023000  004737  030246                  JSR     PC,$SB2D        ;CONVERT IT
023004  004537  027556                  JSR     R5,REPLZ        ;TYPE IT
023010  000003                          .WORD   3               ;TYPE 3 DIGITS
023012  104401  052716                  TYPE    ,LINSP          ;SPACES
023016  010246                          MOV     R2,-(SP)        ;PUT $OPERC ON THE STACK
023020  004737  033546                  JSR     PC,$DB2D        ;CONVERT IT
023024  004537  027556                  JSR     R5,REPLZ        ;TYPE $OPERC
023030  000006                          .WORD   6               ;TYPE 6 DIGITS
023032  104401  052716                  TYPE    ,LINSP          ;SPACES
023036  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023042  010246                          MOV     R2,-(SP)        ;PUT $POSIT ON THE STACK
023044  004737  033546                  JSR     PC,$DB2D        ;CONVERT IT
023050  004537  027556                  JSR     R5,REPLZ        ;TYPE $POSIT
023054  000006                          .WORD   6               ;TYPE 6 DIGITS
023056  104401  052716                  TYPE    ,LINSP          ;SPACES
023062  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023066  010246                          MOV     R2,-(SP)        ;PUT $TRANS ON THE STACK
023070  004737  033546                  JSR     PC,$DB2D        ;CONVERT $TRANS
023074  004537  027556                  JSR     R5,REPLZ        ;TYPE IT
023100  000012                          .WORD   10.             ;TYPE 10 DIGITS
023102  104401  052716                  TYPE    ,LINSP          ;SPACES
023106  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023112  010246                          MOV     R2,-(SP)        ;PUT $READ ON THE STACK
023114  004737  033546                  JSR     PC,$DB2D        ;CONVERT $READ
023120  004537  027556                  JSR     R5,REPLZ        ;TYPE IT
023124  000012                          .WORD   10.             ;TYPE 10 DIGITS
023126  104401  052717                  TYPE    ,LINSPO         ;1 SPACE
023132  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023136  062702  000002                  ADD     #2,R2           ;INCREMENT R2 AGAIN
023142  012246                          MOV     (R2)+,-(SP)     ;PUT $SOFT ON THE STACK
023144  004737  030246                  JSR     PC,$SB2D        ;CONVERT $SOFT
023150  004537  027556                  JSR     R5,REPLZ        ;TYPEOUT $SOFT
023154  000004                          .WORD   4               ;TYPE 4 DIGITS
023156  104401  052717                  TYPE    ,LINSPO         ;SPACES
023162  012246                          MOV     (R2)+,-(SP)     ;PUT $HARD ON THE STACK
023164  004737  030246                  JSR     PC,$SB2D        ;CONVERT $HARD
023170  004537  027556                  JSR     R5,REPLZ        ;TYPEOUT $HARD
023174  000004                          .WORD   4               ;TYPE 4 DIGITS
023176  104401  052717                  TYPE    ,LINSPO         ;SPACES
023202  012246                          MOV     (R2)+,-(SP)     ;PUT $SKI ON THE STACK
```

```
023204  004737  030246                  JSR     PC,$SB2D        ;CONVERT $SKI
023210  004537  027556                  JSR     R5,REPLZ        ;TYPEOUT $SKI
023214  000004                          .WORD   4               ;TYPE 4 DIGITS
023216  104401  052717                  TYPE    ,LINSPO         ;SPACES
023222  012246                          MOV     (R2)+,-(SP)     ;PUT $MISPO ON THE STACK
023224  004737  030246                  JSR     PC,$SB2D        ;CONVERT $MISPO
023230  004537  027556                  JSR     R5,REPLZ        ;TYPEOUT $MISPO
023234  000004                          .WORD   4               ;TYPE 4 DIGITS
023236  104401  052717                  TYPE    ,LINSPO         ;SPACES
023242  016046  000056                  MOV     $TOTAL(R0),-(SP)  ;CALCULATE NUMBER OF OTHER ERRORS
023246  166016  000060                  SUB     $SOFT(R0),(SP)  ;SUBTRACT $SOFT FROM $TOTAL
023252  166016  000062                  SUB     $HARD(R0),(SP)  ;SUBTRACT $HARD FROM $TOTAL
023256  166016  000064                  SUB     $SKI(R0),(SP)   ;SUBTRACT $SKI FROM $TOTAL
023262  166016  000066                  SUB     $MISPO(R0),(SP) ;SUBTRACT $MISPO FROM $TOTAL
023266  004737  030246                  JSR     PC,$SB2D        ;CONVERT 'OTHER' COUNT
023272  004537  027556                  JSR     R5,REPLZ        ;TYPE IT
023276  000004                          .WORD   4               ;TYPE 4 DIGITS
023300  005726                          TST     (SP)+           ;CLEAR STACK
023302  000207                          RTS     PC


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

023304  005737  001362          INCSOF: TST     BADSEC          ;SEE IF BAD TRK/SEC INDICATOR SET
023310  001006                          BNE     1$              ;BR IF IT'S SET, DON'T INCREMENT COUNT
023312  026027  000060  023417          CMP     $SOFT(R0),#9999.  ;IS $SOFT ALREADY AT MAXIMUM ?
023320  103002                          BHIS    1$              ;BR IF IT IS
023322  005260  000060                  INC     $SOFT(R0)               ;INCREMENT $SOFT
023326  000207                  1$:     RTS     PC              ;RETURN


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

023330  005737  001362          INCHRD: TST     BADSEC          ;SEE IF BAD TRK/SEC INDICATOR SET
023334  001006                          BNE     1$              ;BR IF IT'S SET, DON'T INCREMENT COUNT
023336  026027  000062  023417          CMP     $HARD(R0),#9999.  ;IS $HARD ALREADY AT MAXIMUM ?
023344  103002                          BHIS    1$              ;BR IF IT IS
023346  005260  000062                  INC     $HARD(R0)               ;INCREMENT $HARD
023352  000207                  1$:     RTS     PC              ;RETURN


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

023354  005737  001362          INCSKI: TST     BADSEC          ;SEE IF BAD TRK/SEC INDICATOR SET
023360  001006                          BNE     1$              ;BR IF IT'S SET, DON'T INCREMENT COUNT
023362  026027  000064  023417          CMP     $SKI(R0),#9999.  ;IS $SKI ALREADY AT MAXIMUM ?
023370  103002                          BHIS    1$              ;BR IF IT IS
023372  005260  000064                  INC     $SKI(R0)                ;INCREMENT $SKI
023376  000207                  1$:     RTS     PC              ;RETURN
```

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

023400  005737  001362                  INCMIS: TST     BADSEC          ;SEE IF BAD TRK/SEC INDICATOR SET
023404  001006                                  BNE     1$              ;BR IF IT'S SET, DON'T INCREMENT COUNT
023406  026027  000066  023417                  CMP     $MISPO(R0),#9999.  ;IS $MISPO ALREADY AT MAXIMUM ?
023414  103002                                  BHIS    1$              ;BR IF IT IS
023416  005260  000066                          INC     $MISPO(R0)              ;INCREMENT $MISPO
023422  000207                          1$:     RTS     PC              ;RETURN


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

023424  005737  001362                  INCTOT: TST     BADSEC          ;SEE IF BAD TRK/SEC INDICATOR SET
023430  001006                                  BNE     1$              ;BR IF IT'S SET, DON'T INCREMENT COUNT
023432  026027  000056  023417                  CMP     $TOTAL(R0),#9999.  ;IS $TOTAL ALREADY AT MAXIMUM ?
023440  103002                                  BHIS    1$              ;BR IF IT IS
023442  005260  000056                          INC     $TOTAL(R0)              ;INCREMENT $TOTAL
023446  000207                          1$:     RTS     PC              ;RETURN


                                        ;ROUTINE TO TYPE THE TIME

023450  005737  001310                  $TIME:  TST     CLKFLG          ;CLOCK ON THE SYSTEM ?
023454  001033                                  BNE     1$              ;BR IF NOT
023456  104401  001201                          TYPE    ,$CRLF          ;CR-LF
023462  013746  001364                          MOV     HOUR,-(SP)      ;PUT 'HOURS' ON THE STACK
023466  004737  030246                          JSR     PC,$SB2D        ;CONVERT TO DECIMAL
023472  004537  027556                          JSR     R5,REPLZ        ;TYPE IT
023476  000002                                  .WORD   2               ;TYPE 2 DIGITS
023500  104401  053766                          TYPE    ,COLON          ;':'
023504  013746  001366                          MOV     MINUTE,-(SP)    ;PUT 'MINUTES' ON THE STACK
023510  004737  030246                          JSR     PC,$SB2D        ;CONVERT TO DECIMAL
023514  004537  027556                          JSR     R5,REPLZ        ;TYPE IT
023520  000002                                  .WORD   2               ;TYPE 2 DIGITS
023522  104401  053766                          TYPE    ,COLON          ;':'
023526  013746  001370                          MOV     SECOND,-(SP)    ;PUT SECONDS ON THE STACK
023532  004737  030246                          JSR     PC,$SB2D        ;CONVERT TO DECIMAL
023536  004537  027556                          JSR     R5,REPLZ        ;TYPE IT
023542  000002                                  .WORD   2               ;TYPE 2 DIGITS
023544  000207                          1$:     RTS     PC


                                        ;CLOCK HANDLER ROUTINE

023546  005337  001372                  CLOCK:  DEC     SIXTEE          ;INCREMENT THE 1/60 SECOND COUNTER
023552  001035                                  BNE     1$              ;BR IF A SECOND NOT COUNTED
023554  013737  001312  001372                  MOV     HZ,SIXTEE       ;RESTORE THE VALUE
023562  005237  001370                          INC     SECOND          ;COUNT THE SECOND
023566  022737  000074  001370                  CMP     #60.,SECOND     ;AT MAXIMUM ?
023574  001024                                  BNE     1$              ;BR IF NOT
023576  005037  001370                          CLR     SECOND          ;CLEAR THE SECOND'S COUNTER
```

```
023602  005237  001470                  INC     INTRVL+2          ;COUNT THE PERFORMANCE SUMMARY INTERVAL
023606  005237  001366                  INC     MINUTE            ;COUNT THE MINUTE
023612  022737  000074  001366          CMP     #60.,MINUTE       ;AT MAXIMUM ?
023620  001012                          BNE     1$                ;BR IF NOT
023622  005037  001366                  CLR     MINUTE            ;CLEAR THE MINUTE'S COUNTER
023626  005237  001364                  INC     HOUR              ;COUNT THE HOURS
023632  022737  001747  001364          CMP     #999.,HOUR        ;AT MAXIMUM
023640  103002                          BHIS    1$                ;BR IF NOT
023642  005037  001364                  CLR     HOUR              ;CLEAR THE HOURS
023646  012746  000020          1$:     MOV     #20,-(SP)         ;1 MS ON THE STACK
023652  004737  040576                  JSR     PC,RPTMR          ;DRIVER TIMER ROUTINE
023656  005737  001466                  TST     INTRVL            ;DISPLAY THE PERFORMANCE SUMMARY ?
023662  001411                          BEQ     2$                ;BR IF NOT
023664  023737  001466  001470          CMP     INTRVL,INTRVL+2   ;DISPLAY INTERVAL FINISHED ?
023672  001005                          BNE     2$                ;BR IF NOT
023674  012737  177777  001314          MOV     #-1,STATIN        ;SET PERFORMANCE SUMMARY DISPLAY FLAG
023702  005037  001470                  CLR     INTRVL+2          ;CLEAR THE PERFORMANCE INTERVAL COUNTER
023706  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

023710  005737  001540          KSR:    TST     ORDERQ+16                 ;ANY OPERATIONS ACTIVE ?
023714  001402                          BEQ     KSR1
023716  104401  054077                  TYPE    ,BUSY             ;'SYSTEM BUSY...'
023722  104412                  KSR1:   SAVREG                    ;SAVE THE REGISTERS
023724  012737  000200  177776          MOV     #PR4,PS           ;SET PRIORITY TO 4
023732  005037  001360                  CLR     CFLAG             ;CLEAR THE 'CONTROL C' FLAG
023736  004737  023450                  JSR     PC,$TIME          ;TYPE THE TIME
023742  005777  155214                  TST     @$TKB             ;CLEAR ANY GARBAGE IN THE TTY BUFFER
023746  104401  053612                  TYPE    ,ENTCOM           ;'ENTER COMMANDS'
023752  104411                          RDLIN                     ;READ THE KEYBOARD
023754  012605                          MOV     (SP)+,R5          ;GET ADDRESS OF INPUT STRING
023756  005737  001360                  TST     CFLAG             ;CHECK THE CONTROL C FLAG
023762  001116                          BNE     7$                ;EXIT IF 'CONTROL C' ENTERED
023764  005205                          INC     R5                ;POINT TO SECOND CHARACTER
023766  122715  000124                  CMPB    #'T,(R5)          ;EQ TO A 'T' ?
023772  001462                          BEQ     8$                ;YES
023774  122715  000101                  CMPB    #'A,(R5)          ;EQ TO AN 'A'
024000  001410                          BEQ     1$                ;BR IF IT IS
024002  121527  000067                  CMPB    (R5),#'7          ;DRIVE NUMBER GREATER THAN AN ASCII 7 ?
024006  101102                          BHI     6$                ;BR IF IT IS
024010  121527  000060                  CMPB    (R5),#'0          ;DRIVE NUMBER LESS THAN AN ASCII 0 ?
024014  103477                          BLO     6$                ;BR IF IT IS
024016  142715  177770                  BICB    #^C7,(R5)         ;LEAVE ONLY LOWER 3 BITS IF CHAR NOT 'A'
024022  122765  000124  177777  1$:     CMPB    #'T,-1(R5)        ;EQ TO 'T'
024030  001003                          BNE     2$                ;BR IF NOT EQ
024032  004737  024576                  JSR     PC,NEWASN         ;ASSIGN DRIVE FOR TEST
024036  000470                          BR      7$                ;EXIT
024040  122765  000104  177777  2$:     CMPB    #'D,-1(R5)        ;EQ TO 'D' ?
024046  001003                          BNE     3$                ;BR IF NOT EQ
```

```
024050  004737  024606                   JSR     PC,DEASGN        ;DEASSIGN DRIVE
024054  000461                           BR      7$               ;EXIT
024056  122765  000123  177777  3$:      CMPB    #'S,-1(R5)       ;EQ TO 'S'
024064  001003                           BNE     4$               ;BR IF NOT EQ
024066  004737  024716                   JSR     PC,SCMND         ;TYPE STATISTICS
024072  000452                           BR      7$               ;EXIT
024074  122765  000127  177777  4$:      CMPB    #'W,-1(R5)       ;EQ TO 'W'
024102  001007                           BNE     5$               ;BR IF NOT EQ
024104  032777  000001  155042           BIT     #SW0,@SWR        ;IS SWITCH 0 SET ?
024112  001040                           BNE     6$               ;BR IF SET, CAN'T DO 'W' COMMAND
024114  004737  025150                   JSR     PC,DATAPK        ;WRITE A DATA PACK
024120  000437                           BR      7$               ;EXIT
024122  122765  000122  177777  5$:      CMPB    #'R,-1(R5)       ;EQ TO 'R' ?
024130  001031                           BNE     6$               ;BR IF NOT EQ
024132  004737  025200                   JSR     PC,REDAPK        ;READ A DATA PACK
024136  000430                           BR      7$               ;EXIT
024140  122765  000127  177777  8$:      CMPB    #'W,-1(R5)       ;WT COMMAND ?
024146  001022                           BNE     6$               ;NO
024150  122765  000101  000001           CMPB    #'A,1(R5)        ;ALL DRIVES ?
024156  001413                           BEQ     9$               ;YES
024160  126527  000001  000067           CMPB    1(R5),#'7        ;GREAT THAN 7
024166  101012                           BHI     6$               ;YES
024170  126527  000001  000000           CMPB    1(R5),#0         ;LESS THAN 0
024176  103406                           BLO     6$               ;YES
024200  142765  177770  000001           BICB    #^C7,1(R5)       ;CHOP OFF THE HIGHER BITS
024206  004737  025162          9$:      JSR     PC,WATPAK        ;ASSIGN DRIVES WITH WT COMMAND
024212  000402                           BR      7$
024214  104401  053570          6$:      TYPE    ,INVLD           ;TYPE 'INVALID COMMAND' MESSAGE
024220  104413                  7$:      RESREG                   ;RESTORE R0 - R5
024222  005777  154734                   TST     @$TKB            ;CLEAR THE TTY BUFFER
024226  052777  000100  154724           BIS     #BIT06,@$TKS     ;SET TTY INTERRUPT ENABLE
024234  005037  177776                   CLR     PS               ;SET PRIORITY BACK TO ZERO
024240  000207                           RTS     PC               ;RETURN

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

024242  122715  000101          ASSIGN:  CMPB    #'A,(R5)         ;ASSIGN ALL DRIVES?
024246  001426                           BEQ     ASGN2            ;BR IF ALL DRIVES
024250  111504                  ASGN1:   MOVB    (R5),R4          ;PUT DRIVE # IN R4
024252  012737  053026  026564           MOV     #UNTASN,ASNMSG   ;'DRIVE ASSIGNED' MESSAGE ADDRESS
024260  136437  034304  001544           BITB    ATABIT(R4),ASNLST  ;DRIVE ALREADY ASSIGNED ?
024266  001014                           BNE     2$               ;BR IF IT IS
024270  005704                           TST     R4               ;TRYING TO ASSIGN DRIVE 0 ?
024272  001007                           BNE     1$               ;BR IF NOT
024274  012737  053106  026564           MOV     #NOTAVL,ASNMSG   ;'NOT AVAILABLE' MESSAGE ADDRESS
024302  122737  000024  000041           CMPB    #24,41           ;SEE IF LOADED FROM AN RM03
024310  001403                           BEQ     2$               ;BR IF RM03 IS THE LOAD DEVICE
024312  004737  024406          1$:      JSR     PC,ASGN3         ;SEE IF DRIVE ON THE SYSTEM
024316  000207                           RTS     PC               ;RETURN
024320  000137  026544          2$:      JMP     ASNERR           ;EXIT ERROR
024324  122737  000024  000041  ASGN2:   CMPB    #24,41           ;LOADED FROM AN RM03 ?
024332  001003                           BNE     1$               ;BR IF NOT
024334  012704  000001                   MOV     #1,R4            ;START WITH DRIVE 1
024340  000401                           BR      2$               ;CONTINUE
024342  005004                  1$:      CLR     R4               ;START WITH DRIVE 0
024344  012737  053026  026564  2$:      MOV     #UNTASN,ASNMSG   ;ERROR MESSAGE
```

```
024352  136437  034304  001544          BITB    ATABIT(R4),ASNLST ;ALREADY ASSIGNED ?
024360  001007                          BNE     4$                ;YES
024362  004737  024406                  JSR     PC,ASGN3          ;ASSIGN THE DRIVE
024366  005204                  3$:     INC     R4                ;INCREMENT DRIVE #
024370  022704  000007                  CMP     #7,R4             ;ALL DRIVE CHECKED ?
024374  002363                          BGE     2$                ;NO
024376  000207                          RTS     PC                ;YES
024400  004737  026544          4$:     JSR     PC,ASNERR         ;ERROR MESSAGE
024404  000770                          BR      3$                ;TO LOOP
024406  136437  034304  001544  ASGN3:  BITB    ATABIT(R4),ASNLST  ;DRIVE ALREADY ASSIGNED ?
024414  001041                          BNE     ASGN4             ;BR IF IT IS
                                ;1$:    TST     DTUW              ;DATA TRANSFER UNDER WAY ?
                                ;       BPL     1$                ;BR IF IT IS
024416  110437  045532                  MOVB    R4,GENDPB         ;DRIVE NUMBER
024422  004737  014770                  JSR     PC,RECALO         ;RECALIBRATE DRIVE
024426  105764  034170                  TSTB    DRVSTA(R4)        ;DRIVE AVAILABLE?
024432  001440                          BEQ     ASGN7             ;BR IF DRIVE OFFLINE OR NONEXISTENT
024434  100432                          BMI     ASGN6             ;BR IF DRIVE UNSAFE
024436  006304                          ASL     R4                ;MAKE R4 INTO WORD INDEX
                                ;       MOV     BLKADR(R4),NEWUNT(R4) ;DPB ADDRESS
024440  016400  002102                  MOV     BLKADR(R4),R0     ;PUT BLOCK'S ADDR INTO R0
024444  004737  025212                  JSR     PC,CLRDPB         ;CLEAR BLOCK FOR DRIVE JUST ASSIGNED
024450  004737  025412                  JSR     PC,DRVPRM         ;GET THE DRIVE'S ADDRESS LIMITS
024454  004737  025654                  JSR     PC,GETID          ;GET DRIVE I.D.
024460  004537  025764                  JSR     R5,GETADR         ;GET BAD SECTOR ADDRESSES
024464  000414                          BR      2$                ;BRANCH IF RETRIEVE FAILS
024466  016464  002102  001570          MOV     BLKADR(R4),NEWUNT(R4) ;DPB ADDRESS
024474  012760  000001  000070          MOV     #1,$PASSC(R0)     ;PRESET PASS COUNT TO 1
024502  005737  001316                  TST     PACK              ;WRITE DATA PACK ?
024506  001403                          BEQ     2$                ;BR IF NOT
024510  113760  001316  000026          MOVB    PACK,$PACK(R0)    ;SET READ/WRITE DATA PACK INDICATOR
024516  006204                  2$:     ASR     R4                ;RESTORE DRIVE ADDRESS
024520  000207                  ASGN4:  RTS     PC                ;RETURN
024522  012737  053125  026564  ASGN6:  MOV     #NOTSAF,ASNMSG    ;'UNSAFE' MESSAGE ADDRESS
024530  000137  026544                  JMP     ASNERR            ;TO ERROR ROUTINE
024534  105764  034200          ASGN7:  TSTB    DRVTYP(R4)        ;DRIVE PRESENT?
024540  001405                          BEQ     1$                ;BR IF NOT
024542  100010                          BPL     2$                ;BR IF DRIVE OFFLINE
024544  012737  053054  026564          MOV     #NOTRM,ASNMSG     ;ADDRESS OF 'NOT RM03' MSG
024552  000407                          BR      3$                ;EXIT
024554  012737  053071  026564  1$:     MOV     #NOTPRS,ASNMSG    ;ADDRESS OF 'NOT PRESENT' MSG
024562  000403                          BR      3$                ;EXIT
024564  012737  052763  026564  2$:     MOV     #UNTOFF,ASNMSG    ;ADDRESS OF 'DRIVE OFFLINE' MESSAGE
                                ;3$:    RTS     PC                ;ERROR RETURN
024572  000137  026544          3$:     JMP     ASNERR            ;TO EPROR ROUTINE

                                ;'T' COMMAND (ROUTINE TO ASSIGN A DRIVE)

024576  005037  001316          NEWASN: CLR     PACK              ;CLEAR 'W' COMMAND INDICATOR
024602  000137  024242                  JMP     ASSIGN            ;GO TO THE ASSIGN ROUTINE

                                ;'D' COMMAND (ROUTINE TO DEASSIGN A DRIVE)

024606  005004                  DEASGN: CLR     R4
024610  012702  000010                  MOV     #8.,R3            ;COUNTER
024614  122715  000101                  CMPB    #'A,(R5)          ;DEASSIGN ALL DRIVES ?
```

```
024620  001403                          BEQ     1$                      ;BR IF YES
024622  012703  000001                  MOV     #BIT00,R3               ;SET R3 FOR ONE UNIT
024626  111504                          MOVB    (R5),R4                 ;GET DRIVE NUMBER
024630  136437  034304  001544  1$:     BITB    ATABIT(R4),ASNLST       ;DRIVE ASSIGNED ?
024636  001414                          BEQ     3$                      ;BR IF NOT
024640  146437  034304  001544          BICB    ATABIT(R4),ASNLST       ;DELETE THE DRIVE FROM THE ASSIGNED LIST
024646  006304                          ASL     R4                      ;MAKE ADDR INTO A WORD INDEX
024650  016464  002102  001546          MOV     BLKADR(R4),DUNIT(R4)    ;PUT ADDRESS IN DEASSIGN LIST
024656  006204                          ASR     R4
024660  005303                  2$:     DEC     R3                      ;ANY MORE DRIVES ?
024662  001412                          BEQ     4$                      ;BR IF NOT
024664  005204                          INC     R4
024666  000760                          BR      1$
024670  012737  053004  026564  3$:     MOV     #UNTNOT,ASNMSG          ;ADDR OF 'NOT ASSIGNED' MESSAGE
024676  004737  026544                  JSR     PC,ASNERR               ;REPORT IT
024702  122715  000101                  CMPB    #'A,(R5)                ;ALL DRIVE ?
024706  001764                          BEQ     2$                      ;YES
024710  104401  001201          4$:     TYPE    ,$CRLF
024714  000207                          RTS     PC

                                ;'S' COMMAND (ROUTINE TO TYPE DRIVE PERFORMANCE SUMMARY)

024716  005004                  SCMND:  CLR     R4
024720  012703  000010                  MOV     #8.,R3                  ;COUNTER
024724  122715  000101                  CMPB    #'A,(R5)                ;ALL STATISTICS ?
024730  001421                          BEQ     3$                      ;BR IF YES
024732  111504                          MOVB    (R5),R4                 ;GET DRIVE NUMBER
024734  136437  034304  001544          BITB    ATABIT(R4),ASNLST       ;SEE IF DRIVE ASSIGNED
024742  001406                          BEQ     1$                      ;BR IF NOT
024744  006304                          ASL     R4                      ;MAKE DRIVE ADDR INTO WORD INDEX
024746  016400  002102                  MOV     BLKADR(R4),R0           ;ADDR OF BLOCK
024752  004737  022702                  JSR     PC,TYPEST               ;TYPE DRIVE STATISTICS
024756  000471                          BR      8$                      ;EXIT
024760  012737  053004  026564  1$:     MOV     #UNTNOT,ASNMSG          ;ADDR OF 'NOT ASSIGNED' MSG
024766  004737  026544                  JSR     PC,ASNERR               ;TYPE ERROR MESSAGE
024772  000463                          BR      8$                      ;EXIT
024774  105737  001544          3$:     TSTB    ASNLST                  ;ANY DRIVE ASSIGNED ?
025000  001001                          BNE     4$                      ;YES
025002  000457                          BR      8$                      ;RETURN
025004  004737  022614          4$:     JSR     PC,STATPR               ;TYPE ALL STATISTICS
025010  105737  001320                  TSTB    DATE                    ;SEE IF 'DATE' ENTERED
025014  001404                          BEQ     11$                     ;BR IF NOT
025016  104401  053770                  TYPE    ,DATEIS                 ;'DATE: '
025022  104401  001320                  TYPE    ,DATE                   ;THE OPERATOR ENTERED DATE
025026  105737  001332          11$:    TSTB    OPERID                  ;SEE IF OPERATOR I.D. ENTERED
025032  001404                          BEQ     12$                     ;BR IF NOT
025034  104401  054001                  TYPE    ,IDIS                   ;'OPERATOR I.D.: '
025040  104401  001332                  TYPE    ,OPERID                 ;THE OPERATOR I.D.
025044  104401  054023          12$:    TYPE    ,HEDLIN                 ;HEADER LINE
025050  012737  042716  025116          MOV     #DRIVED+$DRVID,6$       ;DRIVE I.D. FIELD ADDRESS
025056  136437  034304  001544  5$:     BITB    ATABIT(R4),ASNLST       ;SEE IF DRIVE ASSIGNED
025064  001417                          BEQ     7$                      ;BR IF NOT ASSIGNED
025066  010446                          MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
025070  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
025072     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
```

```
025073      000                          .BYTE   0              ;;SUPPRESS LEADING ZEROS
025074  104401  052714                   TYPE    ,LIN4SP        ;4 SPACES
025100  105777  000012                   TSTB    @6$            ;SEE IF DRIVE I.D. ENTERED
025104  001003                           BNE     10$            ;BR IF DRIVE I.D. PRESENT
025106  104401  054046                   TYPE    ,NONE          ;TYPE 'NONE'
025112  000404                           BR      7$             ;CONTINUE
025114  104401                   10$:    TYPE                   ;TYPE THE DRIVE I.D.
025116  000000                   6$:     .WORD   0              ;ADDRESS OF DRIVE I.D. FIELD HERE
025120  104401  001201                   TYPE    ,$CRLF         ;CR-LF
025124  005303                   7$:     DEC     R3             ;DECREMENT THE COUNTER
025126  003405                           BLE     8$             ;BR IF AT END
025130  062737  000304  025116           ADD     #$RMEC2+2,6$   ;INCREMENT THE MESSAGE FIELD ADDRESS
025136  005204                           INC     R4             ;INCREMENT DRIVE ADDRESS
025140  000746                           BR      5$             ;CONTINUE
025142  104401  001201           8$:     TYPE    ,$CRLF         ;CR-LF
025146  000207                   9$:     RTS     PC

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

025150  012737  177777  001316   DATAPK: MOV     #-1,PACK       ;SET THE 'W' COMMAND INDICATOR
025156  000137  024242                   JMP     ASSIGN         ;ASSIGN REQUESTED DRIVE

                                 ;'WT' COMMAND (TO WRITE A PACK AND FOLLOWED BY TEST PACK)
025162  116515  000001           WATPAK: MOVB    1(R5),(R5)     ;ADJUST DRIVE NUMBER ADDRESS
025166  012737  177776  001316           MOV     #-2,PACK       ;PACK WRITE COMMAND
025174  000137  024242                   JMP     ASSIGN         ;JUMP TO ASSIGN ROUTINE
                                 ;'R' COMMAND (ROUTINE TO READ A DATA PACK)

025200  012737  000001  001316   REDAPK: MOV     #1,PACK        ;SET THE 'READ' INDICATOR
025206  000137  024242                   JMP     ASSIGN         ;ASSIGN THE REQUESTED DRIVE

                                 ;ROUTINE TO CLEAR THE DPB FOR THE ASSIGNED DRIVE
                                 ;CALL:
                                 ;       MOV     #DPB,R0        ;DPB ADDRESS
                                 ;       JSR     PC,CLRDPB
                                 ;       RETURN

025212                           CLRDPB:
025212  010146                           MOV     R1,-(SP)       ;;PUSH R1 ON STACK
025214  010346                           MOV     R3,-(SP)       ;;PUSH R3 ON STACK
025216  010446                           MOV     R4,-(SP)       ;;PUSH R4 ON STACK
025220  010546                           MOV     R5,-(SP)       ;;PUSH R5 ON STACK
025222  010004                           MOV     R0,R4          ;GET THE DPB ADDRESS
025224  062704  000002                   ADD     #2,R4          ;ADDRESS OF FIRST LOCN TO BE CLEARED
025230  012703  000005                   MOV     #5,R3          ;NUMBER OF LOCNS TO BE CLEARED
025234  005024                   1$:     CLR     (R4)+          ;CLEAR THE LOCATION
025236  005303                           DEC     R3             ;DECREMENT THE COUNTER
025240  001375                           BNE     1$             ;BR IF NOT FINISHED
025242  062704  000002                   ADD     #2,R4          ;MOVE THE ADDRESS PAST THE 'REG' ADDR
025246  012703  000070                   MOV     #$NEXT-$REG,R3 ;NUMBER OF LOCNS TO BE CLEARED
025252  005024                   2$:     CLR     (R4)+          ;CLEAR
025254  162703  000002                   SUB     #2,R3          ;DECREMENT THE LOCN COUNTER
025260  001374                           BNE     2$             ;BR IF NOT FINISHED
025262  062704  000014                   ADD     #12.,R4        ;MOVE PAST ADDRESS LIMITS
025266  012703  000162                   MOV     #$RMEC2-MINSEC,R3  ;NUMBER OF LOCNS TO BE CLEARED
025272  005024                   3$:     CLR     (R4)+          ;CLEAR A LOCATION
```

```
025274  162703  000002                  SUB     #2,R3               ;DECREMENT THE COUNTER
025300  001374                          BNE     3$                  ;BR IF NOT DONE
025302  113760  001516  000024          MOVB    BEGCOD,$CODE(R0)    ;INITIAL COMMAND CODE
025310  013701  001516                  MOV     BEGCOD,R1           ;GET THE ACTUAL OP CODE
025314  116160  002122  000002          MOVB    COMTBL(R1),$COMND(R0)  ;OPERATION CODE
025322  113760  001514  000030          MOVB    BEGPAT,$PATTC(R0)   ;PATTERN CODE
025330  106360  000030                  ASLB    $PATTC(R0)          ;CONVERT CODE TO A TABLE INDEX
025334  013760  001520  000020          MOV     BEGSIZ,$WROL(R0)    ;BEGINNING RECORD SIZE
025342  013760  001520  000004          MOV     BEGSIZ,$WROM(R0)    ;VALUE FOR DATA TRANSFER
025350  005460  000004                  NEG     $WROM(R0)           ;MAKE IT INTO 2'S COMPLEMENT
025354  012760  000400  000022          MOV     #256.,$SSEC(R0)     ;INITIAL VALUE OF SECTOR SIZE
025362  132760  000001  000024          BITB    #1,$CODE(R0)        ;HEADER ORDER ?
025370  001403                          BEQ     4$                  ;BR IF NOT
025372  062760  000002  000022          ADD     #2,$SSEC(R0)        ;ADD HEADER SIZE TO SECTOR SIZE
025400                          4$:
025400  012605                          MOV     (SP)+,R5            ;;POP STACK INTO R5
025402  012604                          MOV     (SP)+,R4            ;;POP STACK INTO R4
025404  012603                          MOV     (SP)+,R3            ;;POP STACK INTO R3
025406  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
025410  000207                          RTS     PC                  ;RETURN

                                ;ROUTINE TO GET ADDRESS LIMITS FROM THE OPERATOR

025412  010346                  DRVPRM: MOV     R3,-(SP)            ;SAVE R3
025414  010446                          MOV     R4,-(SP)            ;SAVE R4
025416  005737  000042                  TST     42                  ;RUNNING UNDER MONITOR CONTROL
025422  001024                          BNE     3$                  ;BR IF YES
025424  104401  053664                  TYPE    ,ENTLMT             ;'ENTER ADDRESSES'
025430  006204                          ASR     R4                  ;CONVERT INDEX TO DRIVE NUMBER
025432  010446                          MOV     R4,-(SP)            ;;SAVE R4 FOR TYPEOUT
                                                                    ;;TYPE DRIVE NUMBER
025434  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
025436     002                          .BYTE   2                   ;;TYPE 2 DIGIT(S)
025437     000                          .BYTE   0                   ;;SUPPRESS LEADING ZEROS
025440  104401  054474                  TYPE    ,SLASH              ;'/'
025444  012737  053155  025466          MOV     #RM03A,2$           ;ADDRESS OF 'RM03' MESSAGE
025452  132764  000004  034200          BITB    #BIT02,DRVTYP(R4)   ;RM03 ?
025460  001001                          BNE     1$                  ;BR IF YES
025462  000000                          HALT                        ;NOT RM03 PROGRAM SHOULD NOT HAVE
                                                                    ;REACHED THIS POINT
025464  104401                  1$:     TYPE                        ;TYPE THE MESSAGE WHICH FOLLOWS
025466  000000                  2$:     .WORD   0                   ;MESSAGE ADDRESS
025470  104401  001201                  TYPE    ,$CRLF              ;CR-LF
025474  012737  001465  001446  3$:     MOV     #821.,CYLIMT        ;ASSUME AN RM03
025502  132764  000004  034200          BITB    #BIT02,DRVTYP(R4)   ;MAKE SURE RM03
025510  001001                          BNE     4$                  ;BR IF RM03
025512  000000                          HALT                        ;HALT IF NOT
025514  116460  034200  000262  4$:     MOVB    DRVTYP(R4),$RMDT(R0)    ;INITIATE $RMDT
025522  062760  177777  000122          ADD     #-1,$FIRST(R0)      ;SEE IF FIRST TIME STARTED
025530  103417                          BCS     5$                  ;BR IF NOT
025532  013760  001446  000106          MOV     CYLIMT,MAXCYL(R0)   ;LOAD MAXIMUM CYLINDER
025540  005060  000110                  CLR     MINCYL(R0)          ;CLEAR MINIMUM CYLINDER
025544  013760  001444  000112          MOV     TRKLMT,MAXTRK(R0)   ;LOAD MAXIMUM TRACK
025552  005060  000114                  CLR     MINTRK(R0)          ;CLEAR MINIMUM TRACK
025556  013760  001442  000116          MOV     SECLMT,MAXSEC(R0)   ;LOAD MAXIMUM SECTOR
025564  005060  000120                  CLR     MINSEC(R0)          ;CLEAR MINIMUM SECTOR
```

```
025570  006304                  5$:     ASL     R4                  ;SETUP TO ADDRESS WORDS
025572  016403  054710                  MOV     TABLE(R4),R3        ;PARAMETER TABLE ADDRESS
025576  013763  001446  000002          MOV     CYLIMT,2(R3)        ;LOAD CYLINDER LIMIT FOR LAST CYLINDER
025604  013763  001446  000010          MOV     CYLIMT,10(R3)       ;LOAD CYLINDER LIMIT FOR STARTING CYLINDER
025612  005737  000042                  TST     42                  ;UNDER MONITOR CONTROL ?
025616  001002                          BNE     6$                  ;BR IF YES
025620  004737  026420                  JSR     PC,PARENT           ;GET THE DRIVE'S PARAMETERS
025624  116060  000120  000010  6$:     MOVB    MINSEC(R0),$SEC(R0)  ;INITIAL SECTOR VALUE
025632  116060  000114  000011          MOVB    MINTRK(R0),$TRK(R0)  ;INITIAL TRACK VALUE
025640  016060  000110  000012          MOV     MINCYL(R0),$CYL(R0)  ;INITIAL CYLINDER VALUE
025646  012604                          MOV     (SP)+,R4            ;RESTORE R4
025650  012603                          MOV     (SP)+,R3            ;RESTORE R3
025652  000207                          RTS     PC                  ;RETURN

                                ;ROUTINE TO GET THE DRIVE I.D. FROM THE OPERATOR

025654  010546                  GETID:  MOV     R5,-(SP)            ;SAVE R5
025656  005737  000042                  TST     42                  ;UNDER MONITOR CONTROL ?
025662  001036                          BNE     2$                  ;BR IF NOT
025664  005037  001360          1$:     CLR     CFLAG               ;CLEAR THE 'CONTROL C' FLAG
025670  104401  053637                  TYPE    ,ENTDRV             ;'ENTER DRV I.D.: '
025674  005046                          CLR     -(SP)               ;CLEAR THE STACK
025676  111016                          MOVB    (R0),(SP)           ;PUT THE DRIVE NUMBER ON THE STACK
025700  104403                          TYPOS                       ;TYPE THE DRIVE NUMBER
025702     002                          .BYTE   2                   ;TYPE 2 DIGITS
025703     000                          .BYTE   0                   ;SUPRESS LEADING ZEROS
025704  104401  001201                  TYPE    ,$CRLF              ;CR-LF
025710  104411                          RDLIN                       ;READ THE ENTRY
025712  012605                          MOV     (SP)+,R5            ;GET THE ENTRY ADDRESS
025714  005737  001360                  TST     CFLAG               ;'CONTROL C' ENTERED ?
025720  001361                          BNE     1$                  ;BR IF IT WAS
025722  121527  000056                  CMPB    (R5),#'.            ;PERIOD ENTERED ?
025726  001414                          BEQ     2$                  ;BR IF YES
025730  112560  000224                  MOVB    (R5)+,$DRVID(R0)    ;STORE THE I.D.
025734  112560  000225                  MOVB    (R5)+,$DRVID+1(R0)  ;STORE THE I.D.
025740  112560  000226                  MOVB    (R5)+,$DRVID+2(R0)  ;STORE THE I.D.
025744  112560  000227                  MOVB    (R5)+,$DRVID+3(R0)  ;STORE THE I.D.
025750  112560  000230                  MOVB    (R5)+,$DRVID+4(R0)  ;STORE THE I.D.
025754  112560  000231                  MOVB    (R5)+,$DRVID+5(R0)  ;STORE THE I.D.
025760  012605                  2$:     MOV     (SP)+,R5            ;RESTORE R5
025762  000207                          RTS     PC                  ;RETURN

                                ;ROUTINE TO GET THE ADDRESSES OF ANY BAD SECTORS (UP TO A MAX OF 16)
                                ;CALL SEQ
                                ;       JSR     R5,GETADR
                                ;       RET1                        ERROR RET
                                ;       RET2                        NORMAL RET
                                ;
                                ;RO DPB ADDRESS

025764                          GETADR:
025764  010146                          MOV     R1,-(SP)            ;;PUSH R1 ON STACK
025766  010246                          MOV     R2,-(SP)            ;;PUSH R2 ON STACK
025770  010346                          MOV     R3,-(SP)            ;;PUSH R3 ON STACK
025772  010446                          MOV     R4,-(SP)            ;;PUSH R4 ON STACK
025774  010001                          MOV     R0,R1               ;DPB ADDRESS
```

```
025776  062701  000124                  ADD     #$BDSEC,R1          ;ADDRESS OF BAD SECTOR TABLE
026002  012702  000040                  MOV     #32.,R2             ;WORD CTR
026006  012721  177777          1$:     MOV     #-1,(R1)+           ;INITIALIZE ALL LOCS TO -1
026012  005302                          DEC     R2                  ;DECREMENT WORD CTR
026014  001374                          BNE     1$                  ;BRANCH IF NOT DONE
026016  112737  000171  045534          MOVB    #RDDAT,GENDPB+$COMND  ;READ DATA COMMAND
026024  012737  001466  045544          MOV     #822.,GENDPB+$CYL     ;LAST CYLINDER
026032  112737  000004  045543          MOVB    #4,GENDPB+$TRK        ;LAST TRACK
026040  012737  177400  045536          MOV     #-256.,GENDPB+$WRDM   ;ONE SECTOR WORD CTR
026046  111037  045532                  MOVB    (R0),GENDPB           ;DRIVE NUMBER
026052  005037  001264                  CLR     $CDW1                 ;ERROR FLAG
026056  005046                          CLR     -(SP)                 ;DUMMY PAIR
026060  005046                          CLR     -(SP)                 ;DUMMY PAIR
026062  012746  000012                  MOV     #12,-(SP)             ;STARTING SECTOR 10.
026066  012746  000036                  MOV     #30.,-(SP)            ;LAST SECTOR
026072  012737  000010  001266          MOV     #10,$CDW2             ;LAST SECTOR
026100  105037  045542                  CLRB    GENDPB+$SEC           ;STARTING SECTOR
026104  004037  035040          2$:     JSR     R0,RM03               ;READ CURRENT SECTOR
026110  045532                          GENDPB                        ;BRANCH IF QUEUE FAILS
026112  000774                          BR      2$                    ;
026114  005737  045550          3$:     TST     GENDPB+$STATUS  ;DONE YET ?
026120  001775                          BEQ     3$                    ;BRANCH IF NOT
026122  100013                          BPL     4$                    ;BRANCH IF DONE WITHOUT ERROR
026124  062737  000002  045542          ADD     #2,GENDPB+$SEC        ;INCREMENT TO NEXT 2ND SECTOR
026132  123737  001266  045542          CMPB    $CDW2,GENDPB+$SEC     ;ALL SECTORS ARE TRIED ?
026140  103361                          BHIS    2$                    ;BRANCH IF NOT
026142  052737  000001  001264          BIS     #BIT0,$CDW1           ;FLAG-RETRIEVE FAILS
026150  000451                          BR      9$                    ;NEXT SETS
026152  010001                  4$:     MOV     R0,R1                 ;BAD SECTOR TABLE
026154  062701  000124                  ADD     #$BDSEC,R1            ;BAD SECTOR TABLE ADDRESS
026160  012702  000040                  MOV     #32.,R2               ;WORD CTR OF BAD SECTOR TABLE
026164  012703  056150                  MOV     #CYLDER,R3            ;BUFFER ADDRESS
026170  062703  000010                  ADD     #10,R3                ;STARTS AT THE 5TH WORD
026174  012704  000374                  MOV     #252.,R4              ;WORD CTR FOR BAD SPOT FILE
026200  022713  177777          5$:     CMP     #-1,(R3)              ;END OF BAD SPOT FILE ?
026204  001433                          BEQ     9$                    ;BRANCH IF SO
026206  022711  177777          6$:     CMP     #-1,(R1)              ;ENTRY OF THE TABLE ?
026212  001406                          BEQ     7$                    ;BRANCH IF SO
026214  062701  000004                  ADD     #4,R1                 ;ADJUST TABLE ENTRY
026220  162702  000002                  SUB     #2,R2                 ;DECREMENT WORD CTR
026224  002420                          BLT     8$                    ;OVER 16 BAD SECTOR DETECTED
026226  000767                          BR      6$                    ;LOOP BACK
026230  011311                  7$:     MOV     (R3),(R1)             ;LOAD THE TABLE
026232  016361  000002  000002          MOV     2(R3),2(R1)           ;TRK AND SECTOR
026240  162704  000002                  SUB     #2,R4                 ;DECREMENT WORD CTR FOR FILE
026244  003413                          BLE     9$                    ;BRANCH IF DONE
026246  062703  000004                  ADD     #4,R3                 ;ADJUST FILE ADDRESS
026252  162702  000002                  SUB     #2,R2                 ;DECREMENT TABLE WORD CTR
026256  003403                          BLE     8$                    ;TABLE ENTRY EXHAUSTED
026260  062701  000004                  ADD     #4,R1                 ;ADJUST TABLE ENTRY
026264  000745                          BR      5$                    ;LOOP BACK
026266  052737  000002  001264  8$:     BIS     #BIT1,$CDW1           ;ERROR FLAG-OVER 16 BAD SECTORS
026274  012637  001266          9$:     MOV     (SP)+,$CDW2           ;UPDATE LAST SECTOR
026300  112637  045542                  MOVB    (SP)+,GENDPB+$SEC     ;STARTING SECTOR
026304  001277                          BNE     2$                    ;BRANCH IF NOT DUMMY PAIR
026306  032737  000001  001264          BIT     #BIT0,$CDW1           ;RETRIEVE FAILS ?
```

```
026314  001412                          BEQ     10$                     ;BRANCH IF NOT
026316  104401  054233                  TYPE    ,MERR1                  ;FAILS TO RETRIEVE BAD SPOT FILE
026322  104401  052755                  TYPE    ,UNTMSG                 ;ON DRIVE
026326  111046                          MOVB    (R0),-(SP)              ; #
026330  104403                          TYPOS
026332     002                          .BYTE   2
026333     000                          .BYTE   0
026334  104401  001201                  TYPE    ,$CRLF                  ;CR-LF
026340  000415                          BR      11$                     ;EXIT
026342  032737  000002  001264  10$:    BIT     #BIT1,$CDW1             ;OVER 16 BAD SECTORS ?
026350  001411                          BEQ     11$                     ;BRANCH IF NOT
026352  104401  054301                  TYPE    ,MERR2                  ;OVER 16 BAD SECTORS DETECTED
026356  104401  052755                  TYPE    ,UNTMSG                 ;ON DRIVE
026362  111046                          MOVB    (R0),-(SP)      ;
026364  104403                          TYPOS
026366     002                          .BYTE   2
026367     000                          .BYTE   0
026370  104401  001201                  TYPE    ,$CRLF
026374  005737  001264          11$:    TST     $CDW1                   ;ERROR FLAG SET ?
026400  001002                          BNE     12$                     ;BRANCH IS SET
026402  062705  000002                  ADD     #2,R5                   ;ADJUST FOR NORMAL RET
026406                          12$:
026406  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
026410  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
026412  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
026414  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
026416  000205                          RTS     R5                      ;EXIT

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

026420  010346                  PARENT: MOV     R3,-(SP)        ;SAVE THE PARAMETER TABLE ADDRESS
026422  005037  001360                  CLR     CFLAG           ;CLEAR THE 'CONTROL C' FLAG
026426  012337  026436          1$:     MOV     (R3)+,3$        ;ADDRESS OF PARAMETER NAME
026432  001442                          BEQ     9$              ;BR IF AT END OF TABLE
026434  104401                          TYPE                    ;TYPE THE PARAMETER NAME
026436  000000                  3$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
026440  012302                          MOV     (R3)+,R2        ;MAXIMUM PARAMETER VALUE
026442  012305                          MOV     (R3)+,R5        ;ADDRESS OF PARAMETER
026444  011546                          MOV     (R5),-(SP)      ;CURRENT VALUE OF PARAMETER
026446  104405                          TYPDS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
026450  104401  054474                  TYPE    ,SLASH          ;' / '
026454  104411                          RDLIN                   ;READ THE KEYBOARD
026456  012601                          MOV     (SP)+,R1        ;INPUT ASCII STRING ADDRESS
026460  005737  001360                  TST     CFLAG           ;'CONTROL C' ENTERED ?
026464  001021                          BNE     8$              ;BR IF IT WAS
026466  004537  030110                  JSR     R5,CK.DIG       ;CHECK THE DIGIT(S)
026472  026426                          1$                      ;CARRIAGE RETURN ONLY ENTERED
026474  026540                          9$                      ;PERIOD ONLY ENTERED
026476  026512                          6$                      ;ILLEGAL INPUT
026500  026506                          5$                      ;TERMINATED WITH A CARRIAGE RETURN
026502  026512                          6$                      ;TERMINATED WITH A ","
026504  026524                          7$                      ;TERMINATED WITH A "."
026506  010215                  5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
```

```
026510  000746                          BR      1$              ;GET MORE PARAMETERS
026512  104401  054055          6$:     TYPE    ,BADENT         ;'BAD ENTRY'
026516  162703  000006                  SUB     #6,R3           ;DECREMENT THE TABLE POINTER
026522  000741                          BR      1$              ;TRY AGAIN
026524  010215                  7$:     MOV     R2,(R5)         ;NEW VALUE
026526  000404                          BR      9$              ;EXIT
026530  005037  001360          8$:     CLR     CFLAG           ;CLEAR THE 'CONTROL C' FLAG
026534  011603                          MOV     (SP),R3         ;RELOAD THE PARAMETER TABLE ADDRESS
026536  000733                          BR      1$              ;TRY AGAIN
026540  005726                  9$:     TST     (SP)+           ;CORRECT THE STACK POINTER
026542  000207                          RTS     PC              ;RETURN

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

026544  104401  053566          ASNERR: TYPE    ,QUES           ;QUESTION MARK
026550  104401  052755                  TYPE    ,UNTMSG         ;TYPE 'DRIVE'
026554  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
026556  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
026560     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
026561     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
026562  104401                          TYPE                    ;TYPE SPECIFIC MESSAGE
026564  000000                  ASNMSG: .WORD   0               ;MESSAGE ADDRESS
026566  104401  001201                  TYPE    ,$CRLF
026572  000207                          RTS     PC

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

026574  005004                  DROP:   CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
026576  111004                          MOVB    (R0),R4         ;MOVE DRIVE NUMBER TO R4
026600  146437  034304  001544          BICB    ATABIT(R4),ASNLST  ;REMOVE DRIVE FROM ASSIGNED LIST
026606  006304                          ASL     R4              ;MAKE DRIVE NUMBER INTO A TABLE INDEX
026610  010064  001546                  MOV     R0,DUNIT(R4)    ;PUT DRIVE IN DROP LIST
026614  104401  001201                  TYPE    ,$CRLF
026620  104401  001201                  TYPE    ,$CRLF
026624  104401  053341                  TYPE    ,DROPNG         ;TYPE 'DROPPING DRIVE'
026630  104401  053452                  TYPE    ,DRNUM          ;'DRIVE #'
026634  006204                          ASR     R4              ;DRIVE NUMBER
026636  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
026640  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
026642     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
026643     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
026644  104401  001201                  TYPE    ,$CRLF
026650  105737  001544                  TSTB    ASNLST          ;MORE DRIVES ACTIVE
026654  001006                          BNE     1$              ;YES
026656  005737  000042                  TST     @#42            ;ANY MONITOR ?
026662  001403                          BEQ     1$              ;NO
026664  005726                          TST     (SP)+           ;CLEAR STACK
```

```
026666  000137  027250                  JMP     $GET42          ;GIVE CONTROL TO MONITOR
026672  000207                  1$:     RTS     PC

                                ;ROUTINE TO DEASSIGN DRIVE IF ERRORS BECOMES EXCESSIVE

026674  032777  000020  152252  ABNRML: BIT     #SW04,@SWR      ;SEE IF SWITCH 4 SET
026702  001006                          BNE     1$              ;BR IF IT'S SET
026704  023760  001464  000056          CMP     MAXER,$TOTAL(R0)  ;CHECK TOTAL ERROR VALUE
026712  103002                          BHIS    1$              ;BR IF ERRORS DONOT EXCEED MAX
026714  000137  026574                  JMP     DROP            ;DEASSING THE DRIVE
026720  000207                  1$:     RTS     PC              ;RETURN

                                ;ROUTINE TO CHECK FOR END OF PASS AND END OF TEST

                                .SBTTL  END OF PASS ROUTINE

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

026722                          $EOP:
026722  005737  001506          EOP:    TST     ENDET           ;END OF PASS DETERMINED BY SEEKS OR WORDS ?
026726  001412                          BEQ     EOP1            ;BR IF SEEKS
026730  026037  000054  001452          CMP     $READ+2(R0),ENDCON+2  ;CHECK MSW OF WORDS READ COUNT
026736  101017                          BHI     EOP2            ;BR IF MSW GREATER THAN LIMIT
026740  103527                          BLO     EOPX            ;BR IF MSW LESS THAN LIMIT
026742  026037  000052  001450          CMP     $READ(R0),ENDCON  ;CHECK LSW AGAINST LIMIT
026750  103012                          BHIS    EOP2            ;BR IF EQUAL OR GREATER
026752  000522                          BR      EOPX            ;EXIT
026754  026037  000044  001456  EOP1:   CMP     $POSIT+2(R0),ENDSEK+2  ;CHECK MSW OF SEEK COUNT
026762  101005                          BHI     EOP2            ;BR IF MSW GREATER THAN LIMIT
026764  103515                          BLO     EOPX            ;EXIT IF MSW LESS THAN LIMIT
026766  026037  000042  001454          CMP     $POSIT(R0),ENDSEK  ;CHECK LSW OF SEEK COUNT
026774  103511                          BLO     EOPX            ;EXIT IF LSW LESS THAN LIMIT
026776  104401  001201          EOP2:   TYPE    ,$CRLF          ;CR-LF
027002  104401  053375                  TYPE    ,ENDPAS         ;END OF PASS FOR THE DRIVE
027006  016046  000070                  MOV     $PASSC(R0),-(SP)  ;PUT PASS COUNT ON THE STACK
027012  104405                          TYPOS                   ;CONVERT PASS COUNT TO DECIMAL AND TYPE IT
027014  111037  001344                  MOVB    (R0),UNIT       ;STORE THE DRIVE NUMBER
027020  032777  000020  152126          BIT     #SW04,@SWR      ;SWITCH 4 SET ?
027026  001017                          BNE     1$              ;BR IF SET
027030  026037  000070  001460          CMP     $PASSC(R0),PASCNT  ;SEE IF AT END OF TEST
027036  103413                          BLO     1$              ;BR IF NOT
027040  104401  053411                  TYPE    ,ENDTST         ;TYPE 'END OF TEST'
027044  104401  053452                  TYPE    ,DRNUM          ;'DRIVE #'
027050  013746  001344                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
027054  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
027056     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
027057     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
027060  104401  001201                  TYPE    ,$CRLF          ;CR-LF
027064  000431                          BR      3$              ;DEASSIGN THE DRIVE
027066  104401  053452          1$:     TYPE    ,DRNUM          ;'DRIVE #'
027072  013746  001344                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
```

```
027076  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
027100     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
027101     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
027102  104401  001201                  TYPE    $CRLF           ;CR-LF
027106  004737  022702                  JSR     PC,TYPEST       ;TYPE THE DRIVE'S STATISTICS
027112  010346                          MOV     R3,-(SP)        ;SAVE R3
027114  010446                          MOV     R4,-(SP)        ;SAVE R4
027116  010004                          MOV     R0,R4           ;DRIVE'S BLOCK ADDRESS
027120  062704  000036                  ADD     #$OPERC,R4      ;ADD THE STARTING ADDR OF SECTIONS TO CLEAR
027124  012703  000010                  MOV     #8.,R3          ;NUMBER OF LOCNS TO BE CLEARED
                                                                ;(ERROR COUNTERS NOT CLEARED)
027130  005024                  2$:     CLR     (R4)+           ;CLEAR THE LOCN
027132  005303                          DEC     R3              ;DECREMENT THE LOCATION COUNTER
027134  001375                          BNE     2$              ;BR IF MORE TO GO
027136  012604                          MOV     (SP)+,R4        ;RESTORE R4
027140  012603                          MOV     (SP)+,R3        ;RESTORE R3
027142  005260  000070                  INC     $PASSC(R0)      ;INCREMENT THE PASS COUNT
027146  000424                          BR      EOPX            ;EXIT
027150  104401  001201          3$:     TYPE    $CRLF
027154  005004                          CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
027156  111004                          MOVB    (R0),R4         ;MOVE DRIVE NUMBER
027160  146437  034304  001544          BICB    ATABIT(R4),ASNLST  ;DELETE DRIVE FROM ASSIGNED LIST
027166  006304                          ASL     R4              ;MAKE DRIVE NUMBER INTO TABLE INDEX
027170  010064  001546                  MOV     R0,DUNIT(R4)    ;PUT BLOCK ADDRESS INTO DROP LIST
027174  105737  001544                  TSTB    ASNLST          ;ALL DRIVES ARE DESIGNED ?
027200  001007                          BNE     EOPX            ;BRANCH IF NOT
027202  005237  001214                  INC     $DEVCT          ;INCREMENT DEVICE COUNT
027206  005237  001212                  INC     $PASS           ;INCREMENT THE PASS NUMBER FOR APT
027212  042737  100000  001212          BIC     #100000,$PASS   ;AVIOD NEGATIVE NUMBER
027220  000207                  EOPX:   RTS     PC              ;RETURN
027222  005237  001212                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
027226  042737  100000  001212          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
027234  005327                          DEC     (PC)+           ;;LOOP?
027236  000001                  $EOPCT: .WORD   1
027240  003013                          BGT     $DOAGN          ;;YES
027242  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
027244  000001                  $ENDCT: .WORD   1
027246  027236                          $EOPCT
027250  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
027254  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
027256  000005                          RESET                   ;;CLEAR THE WORLD
027260  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
027262  000240                          NOP                     ;;SAVE ROOM
027264  000240                          NOP                     ;;FOR
027266  000240                          NOP                     ;;ACT11
027270                          $DOAGN:
027270  000137                          JMP     @(PC)+          ;;RETURN
027272  027274                  $RTNAD: .WORD   FISK
027274  005237  001210          FISK:   INC     $TESTN          ;INCREMENT THE TEST NUMBER IN THE MAIL BOX
027300  000137  005524                  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
```

```
027304  013746  033450          GETREM: MOV     $LONUM,-(SP)     ;STORE RANDOM NUMBER ON THE STACK FOR DIVIDE
027310  013746  033446                  MOV     $HINUM,-(SP)     ;UPPER PART
027314  010546                          MOV     R5,-(SP)         ;PUT THE DIVISOR ONTO THE STACK
027316  004737  027332                  JSR     PC,LINKDV        ;DIVIDE THE RANDOM NUMBERS
027322  012605                          MOV     (SP)+,R5         ;PUT THE REMAINDER INTO R5
027324  005726                          TST     (SP)+            ;ADJUST THE STACK POINTER
027326  000240                          NOP                      ;FOR DEBUGGING HALT
027330  000207                          RTS     PC

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

027332  104412                  LINKDV: SAVREG                   ;STORE R0 - R5
027334  016605  000026                  MOV     26(SP),R5        ;DIVISOR
027340  005004                          CLR     R4               ;OTHER DIVISOR WORD
027342  016602  000030                  MOV     30(SP),R2        ;UPPER DIVIDEND WORD
027346  016603  000032                  MOV     32(SP),R3        ;LOWER DIVIDEND WORD
027352  005000                          CLR     R0               ;CLEAR OTHER DIVIDEND REGISTERS
027354  005001                          CLR     R1
027356  004737  027400                  JSR     PC,M.DPID        ;GO TO THE DIVIDE ROUTINE
027362  010166  000030                  MOV     R1,30(SP)        ;REMAINDER ON THE STACK
027366  010366  000032                  MOV     R3,32(SP)        ;QUOTIENT ON THE STACK
027372  104413                          RESREG                   ;RESTORE R0 - R5
027374  012616                          MOV     (SP)+,(SP)       ;MOVE RETURN UP THE STACK
027376  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
                                ;
027400  012746  000040          M.DPID: MOV     #40,-(SP)        ;COUNTER FOR DIVISION CYCLES
027404  010446                          MOV     R4,-(SP)         ;HIGH ORDER
027406  010546                          MOV     R5,-(SP)         ;LOW ORDER DIVISOR TO THE STACK
027410  005466  000002                  NEG     2(SP)            ;FORM NEGATIVE
027414  005416                          NEG     @SP              ;VERSION OF THE DIVISOR
027416  005666  000002                  SBC     2(SP)
027422  061601                          ADD     @SP,R1
027424  005500                          ADC     R0               ;PERFORM THE INITIAL SUBTRACTION
027426  066600  000002                  ADD     2(SP),R0
027432  103445                          BCS     M.DP50           ;IF CARRY THEN OVERFLOW HAS OCCURRED
027434  005046                          CLR     -(SP)            ;THIS IS A LONGER LASTING CARRY BIT
027436  006103                  M.DP40: ROL     R3
027440  006102                          ROL     R2
027442  006101                          ROL     R1
027444  006100                          ROL     R0
027446  005716                          TST     @SP              ;TEST "CARRY" INDICATOR
027450  001410                          BEQ     M.DP41           ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
027452  005016                          CLR     @SP              ;CLEAR UP FOR NEXT TIME
027454  066601  000002                  ADD     2(SP),R1
027460  005500                          ADC     R0               ;ADD -(DIVISOR)
027462  005516                          ADC     @SP     ;  I     ;SET "CARRY"
```

```
027464  066600  000004                  ADD     4(SP),R0;<-
027470  000404                          BR      M.DP42
027472  060501                  M.DP41: ADD     R5,R1
027474  005500                          ADC     R0                      ;ADD +(DIVISOR)
027476  005516                          ADC     @SP         ;  I        ;SET "CARRY"
027500  060400                          ADD     R4,R0       ;<-
027502  005516                  M.DP42: ADC     @SP                     ;SET "CARRY"
027504  005716                          TST     @SP                     ;TEST THE UPDATE INDICATOR
027506  001401                          BEQ     .+4         ;->         ;IF ZERO FORGET IT
027510  005203                          INC     R3          ;  I        ;NO CARRY POSSIBLE HERE
027512  005366  000006                  DEC     6(SP)       ;<-         ;DECREMENT COUNTER
027516  003347                          BGT     M.DP40                  ;BRANCH IF MORE TO DO
027520  006003                          ROR     R3
027522  103404                          BCS     M.DP44
027524  060501                          ADD     R5,R1
027526  005500                          ADC     R0
027530  060400                          ADD     R4,R0
027532  000241                          CLC
027534  006103                  M.DP44: ROL     R3
027536  062706  000010                  ADD     #10,SP                  ;ADJUST STACK BY 4 WORDS
027542  000242                          CLV
027544  000207                          RTS     PC
027546  062706  000006          M.DP50: ADD     #6,SP
027552  000262                          SEV
027554  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

027556  010046                  REPLZ:  MOV     R0,-(SP)                ;SAVE R0
027560  012746  000012                  MOV     #10.,-(SP)              ;MAXIMUM NUMBER OF DIGITS TO BE TYPED
027564  162516                          SUB     (R5)+,(SP)              ;SUBTACT DIGITS TO FORM INDEX
027566  016600  000006                  MOV     6(SP),R0                ;ADDRESS OF NUMBER TO R0
027572  122710  000060          1$:     CMPB    #'0,(R0)                ;BYTE EQUAL TO ASCII '0' ?
027576  001004                          BNE     2$                      ;BR IF NOT
027600  112710  000040                  MOVB    #40,(R0)                ;REPLACE THE ZERO WITH A SPACE
027604  005200                          INC     R0                      ;INCREMENT THE BYTE ADDRESS
027606  000771                          BR      1$                      ;GO BACK AND LOOK FOR MORE LEADING ZEROS
027610  105710                  2$:     TSTB    (R0)                    ;SEE IF ZERO BYTE TERMINATOR
027612  001003                          BNE     3$                      ;BR IF NOT
027614  005300                          DEC     R0                      ;BACKUP STRING POINTER
027616  112710  000060                  MOVB    #'0,(R0)                ;PUT A ZERO BACK IN
027622  016637  000006  027636  3$:     MOV     6(SP),4$                ;PUT ADDRESS IN LOCATION FOR TYPEOUT
027630  062637  027636                  ADD     (SP)+,4$                ;BEGINNING OF SIGNIFICANT DIGITS
027634  104401                          TYPE                            ;TYPE THE NUMBER
027636  000000                  4$:     .WORD   0                       ;ADDRESS OF NUMBER
027640  012600                          MOV     (SP)+,R0                ;RESTORE R0
027642  012616                          MOV     (SP)+,(SP)              ;MOVE RETURN ADDRESS
027644  000205                          RTS     R5                      ;RETURN

                                ;TYPE NUMERICAL ASCIZ STRING SUPRESS LEADING ZEROS
```

```
                                        ;CALL
                                        ;       MOV     #NUMADR,-(SP)   ;FIRST ADDRESS OF ASCIZ STRING
                                        ;       JSR     PC,$SUPRS
027646  010046                          $SUPRS: MOV     R0,-(SP)        ;SAVE R0
027650  016600  000004                          MOV     4(SP),R0        ;PICKUP THE POINTER
027654  105710                          1$:     TSTB    (R0)            ;TERMINATOR ?
027656  001403                                  BEQ     2$              ;BR IF YES
027660  122720  000060                          CMPB    #'0,(R0)+       ;IS THIS AN ASCII '0' ?
027664  001773                                  BEQ     1$              ;BR IF YES
027666  005300                          2$:     DEC     R0              ;BACKUP BY '1'
027670  010037  027676                          MOV     R0,3$           ;SAVE FOR TYPING
027674  104414                                  DISPLY                  ;GO PRINT
027676  000000                          3$:     .WORD   0               ;ASCIZ POINTER GOES HERE
027700  012600                                  MOV     (SP)+,R0        ;RESTORE R0
027702  012616                                  MOV     (SP)+,(SP)      ;RESTORE THE STACK
027704  000207                                  RTS     PC              ;RETURN

                                        ;ROUTINE TO TYPE AT PRIORITY 4

027706  013746  177776                  TYPRI4: MOV     @#PS,-(SP)      ;SAVE THE PRESENT STATUS
027712  012737  000200  177776                  MOV     #200,@#PS       ;CHANGE THE PRIORITY TO 4
027720  012537  027730                          MOV     (R5)+,1$        ;MESSAGE ADDRESS
027724  004737  031326                          JSR     PC,$TYPE        ;TYPE THE MESSAGE
027730  000000                          1$:     .WORD   0               ;MESSAGE ADDRESS GOES HERE
027732  000205                                  RTS     R5              ;RETURN

                                        ;ROUTINE TO TYPE ERRORS
                                        ;CALL
                                        ;       DISPLY                  ;MUST DEFINED IN 'TRAP' TABLE
                                        ;       MESADR                  ;ADDRESS OF MESSAGE
                                        ;       RETURN

027734  032777  020000  151212          $DSPLY: BIT     #BIT13,@SWR     ;INHIBIT ERROR TYPEOUT ?
027742  001004                                  BNE     1$              ;BR IF YES
027744  005037  177776                          CLR     @#PS            ;SET PRIORITY TO ZERO
027750  000137  031326                          JMP     $TYPE           ;TYPE THE MESSAGE
027754  062716  000002                  1$:     ADD     #2,(SP)         ;INCREMENT THE RETURN
027760  000002                                  RTI                     ;RETURN

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

027762  121127  000060                  CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
027766  103407                                  BLO     1$              ;YES -- BRANCH
027770  121127  000067                          CMPB    (R1),#'7        ;GREATER THAN SEVEN?
027774  101004                                  BHI     1$              ;YES -- BRANCH
027776  111102                                  MOVB    (R1),R2         ;GET THE CHARACTER
030000  042702  177770                          BIC     #^C7,R2         ;STRIP AWAY THE ASCII
030004  005725                                  TST     (R5)+           ;ADJUST FOR RETURN
```

```
030006  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

030010  121127  000060          CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
030014  103407                          BLO     1$              ;YES -- BRANCH
030016  121127  000071                  CMPB    (R1),#'9        ;GREATER THAN NINE?
030022  101004                          BHI     1$              ;YES -- BRANCH
030024  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
030026  042702  000060                  BIC     #'0,R2          ;STRIP AWAY THE ASCII
030032  005725                          TST     (R5)+           ;ADJUST FOR RETURN
030034  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

030036  105711                  CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
030040  001417                          BEQ     3$              ;YES -- BRANCH
030042  121127  000054                  CMPB    (R1),#',        ;"COMMA"?
030046  001413                          BEQ     2$              ;YES -- BRANCH
030050  121127  000056                  CMPB    (R1),#'.        ;"PERIOD"?
030054  001407                          BEQ     1$              ;YES -- BRANCH
030056  004537  030010                  JSR     R5,CK.DEC       ;"DIGIT"?
030062  000410                          BR      4$              ;NO -- BRANCH
030064  004537  027762                  JSR     R5,CK.OCT       ;OCTAL ?
030070  005725                          TST     (R5)+           ;DIGIT BETWEEN 8-9
030072  005725                          TST     (R5)+           ;DIGIT BETWEEN 0-7
030074  005725                  1$:     TST     (R5)+           ;PERIOD
030076  005725                  2$:     TST     (R5)+           ;COMMA
030100  005725                  3$:     TST     (R5)+           ;CARRIAGE RETURN
030102  005201                          INC     R1              ;MOVE POINTER TO NEXT CHARACTER
030104  011505                  4$:     MOV     (R5),R5         ;UNKNOWN CHARACTER
030106  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

030110  010446                          CK.DIG: MOV     R4,-(SP)        ;SAVE R4
030112  010346                                  MOV     R3,-(SP)        ;SAVE R3
030114  010246                                  MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
030116  005002                                  CLR     R2              ;START WITH 0
030120  005003                                  CLR     R3
030122  005004                                  CLR     R4
030124  004537  030036                          JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
030130  030224                                  6$                      ;ILLEGAL CHARACTER
030132  030232                                  9$                      ;CARRIAGE RETURN
030134  030224                                  6$                      ;" "
030136  030226                                  7$                      ;"."
030140  030144                                  1$                      ;DIGIT 0-7
030142  030144                                  1$                      ;DIGIT 8-9
030144  062705  000004                  1$:     ADD     #4,R5           ;STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
030150  006303                          2$:     ASL     R3              ;INPUT NUMBER *2
030152  010346                                  MOV     R3,-(SP)        ;SAVE *2
030154  006303                                  ASL     R3              ;*4
030156  006303                                  ASL     R3              ;*8
030160  062603                                  ADD     (SP)+,R3        ;(*2)+(*8) = *10
030162  060203                                  ADD     R2,R3           ;UPDATE THE INPUT NUMBER
030164  004537  030036                          JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
030170  030230                                  8$                      ;ILLEGAL CHARACTER
030172  030214                                  5$                      ;CARRIAGE RETURN
030174  030212                                  4$                      ;" "
030176  030204                                  3$                      ;"."
030200  030150                                  2$                      ;DIGIT 0-7
030202  030150                                  2$                      ;DIGIT 8-9
030204  105711                          3$:     TSTB    (R1)            ;DOES A "CR" FOLLOW THE "PERIOD"
030206  001010                                  BNE     8$              ;BR IF NOT
030210  005724                                  TST     (R4)+           ;INCREMENT THE RETURN
030212  005724                          4$:     TST     (R4)+           ;INCREMENT THE RETURN
030214  005724                          5$:     TST     (R4)+           ;INCREMENT THE RETURN
030216  020316                                  CMP     R3,(SP)         ;CHECK THE MAGNITUDE OF THE NUMBER
030220  101004                                  BHI     9$              ;BR IF ENTERED NUMBER TOO LARGE
030222  000402                                  BR      8$              ;BYPASS INCREMENT
030224  005725                          6$:     TST     (R5)+           ;INCREMENT RETURN PAST INVALID RETURN
030226  005725                          7$:     TST     (R5)+           ;INCREMENT RETURN
030230  060405                          8$:     ADD     R4,R5           ;SETUP RETURN POINTER
030232  010302                          9$:     MOV     R3,R2           ;ENTERED VALUE
030234  005726                                  TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
030236  012603                                  MOV     (SP)+,R3        ;RESTORE R3
030240  012604                                  MOV     (SP)+,R4        ;RESTORE R4
030242  011505                                  MOV     (R5),R5         ;GET RETURN ADDRESS
030244  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

030246  016637  000002  030272  $SB2D:  MOV     2(SP),1$        ;SAVE THE BINARY NUMBER
030254  012746  030272                  MOV     #1$,-(SP)       ;SET THE POINTER
030260  004737  033546                  JSR     PC,$DB2D        ;CALL THE DOUBLE LENGTH CONVERT
030264  012666  000002                  MOV     (SP)+,2(SP)     ;PICKUP THE POINTER
030270  000207                          RTS     PC              ;RETURN
030272  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,$SB20        ;CALL
                                        ;       RETURN                  ;ADDRESS OF THE 1ST ASCIZ CHAR IS ON THE STACK
                                        ;
                                        ;NOTE:  THE PROGRAM REQUIRES THIS FORM OF '$SB20', NOT THE VERSION ON
                                        ;       THE SYSMAC LIBRARY, REV C AND LATER

030276  016637  000002  030322  $SB20:  MOV     2(SP),1$        ;SAVE THE BINARY NUMBER
030304  012746  030322                  MOV     #1$,-(SP)       ;SET THE POINTER
030310  004737  033742                  JSR     PC,$DB20        ;CALL THE DOUBLE LENGTH CONVERT
030314  012666  000002                  MOV     (SP)+,2(SP)     ;PICKUP THE POINTER
030320  000207                          RTS     PC              ;RETURN
030322  000000  000000          1$:     .WORD   0,0

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

030326  012737  030356  000060  $TKINT: MOV     #$TKSRV,TKVEC   ;SETUP VECTOR
030334  012737  000100  000062          MOV     #100,TKVEC+2    ;PRIORITY TO 4
030342  005777  150614                  TST     @$TKB           ;CLEAR THE BUFFER
030346  012777  000100  150604          MOV     #BIT06,@$TKS    ;SET INTERRUPT ENABLE
030354  000207                          RTS     PC              ;RETURN

                                        ;KEYBOARD INTERRUPT SERVICE ROUTINE
                                        ;CALL
                                        ;       ENTER VIA INTERRUPT

030356  104410                  $TKSRV: RDCHR                   ;READ THE KEYBOARD
030360  112637  030506                  MOVB    (SP)+,5$        ;GET THE CHARACTER
030364  023727  030506  000003          CMP     5$,#3           ;'CONTROL C' ?
030372  001012                          BNE     1$              ;BR IF NOT
030374  104401  001201                  TYPE    ,$CRLF          ;CR-LF
030400  104401  031000                  TYPE    ,$CNTLC         ;'↑C'
030404  012737  177777  001360          MOV     #-1,CFLAG       ;SET THE 'CONTROL C' FLAG
030412  005077  150542                  CLR     @$TKS           ;CLEAR THE TTY INTERRUPT
030416  000432                          BR      4$              ;EXIT
030420  023727  001154  000176  1$:     CMP     SWR,#SWREG      ;SOFTWARE SWITCH REGISTER IN USE ?
```

```
030426  001024                          BNE     3$              ;BR IF NOT
030430  023727  030506  000007          CMP     5$,#7           ;'CONTROL G' ?
030436  001020                          BNE     3$              ;BR IF NOT
030440  104401  001201                  TYPE    ,$CRLF          ;CR-LF
030444  104401  033321                  TYPE    ,$CNTLG         ;'↑G'
030450  013746  177776                  MOV     PS,-(SP)        ;PUT THE STATUS WORD ON THE STACK
030454  012746  030470                  MOV     #2$,-(SP)       ;RETURN ADDRESS
030460  005077  150474                  CLR     @$TKS           ;CLEAR THE TTY INTERRUPT ENABLE
030464  000137  032762                  JMP     $GTSWR          ;GET THE SWITCH REGISTER ENTRY
030470  012777  000100  150462  2$:     MOV     #100,@$TKS      ;ENABLE TTY KEYBOARD INTERRUPT
030476  000402                          BR      4$              ;EXIT
030500  104401  030506          3$:     TYPE    ,5$             ;ECHO THE CHARACTER
030504  000002                  4$:     RTI                     ;RETURN

030506  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

030510  010346                  $RDLIN: MOV     R3,-(SP)        ;SAVE R3
030512  005046                          CLR     -(SP)           ;CLEAR THE RUBOUT KEY
030514  012703  030766          1$:     MOV     #$TTYIN,R3      ;GET ADDRESS
030520  022703  031000          2$:     CMP     #$TTYIN+10.,R3  ;BUFFER FULL?
030524  101467                          BLOS    4$              ;BR IF YES
030526  104410                          RDCHR                   ;GO READ ONE CHARACTER FROM THE TTY
030530  112613                          MOVB    (SP)+,(R3)      ;GET CHARACTER
030532  122713  000177                  CMPB    #177,(R3)       ;IS IT A RUBOUT
030536  001022                          BNE     5$              ;BR IF NO
030540  005716                          TST     (SP)            ;IS THIS THE FIRST RUBOUT?
030542  001007                          BNE     6$              ;BR IF NO
030544  112737  000134  030764          MOVB    #'\,9$          ;TYPE A BACK SLASH
030552  104401  030764                  TYPE    ,9$
030556  012716  177777                  MOV     #-1,(SP)        ;SET THE RUBOUT KEY
030562  005303                  6$:     DEC     R3              ;BACKUP BY ONE
030564  020327  030766                  CMP     R3,#$TTYIN      ;STACK EMPTY?
030570  103445                          BLO     4$              ;BR IF YES
030572  111337  030764                  MOVB    (R3),9$         ;SETUP TO TYPEOUT THE DELETED CHAR.
030576  104401  030764                  TYPE    ,9$             ;GO TYPE
030602  000746                          BR      2$              ;GO READ ANOTHER CHAR.
030604  005716                  5$:     TST     (SP)            ;RUBOUT KEY SET?
030606  001406                          BEQ     7$              ;BR IF NO
030610  112737  000134  030764          MOVB    #'\,9$          ;TYPE A BACK SLASH
030616  104401  030764                  TYPE    ,9$
030622  005016                          CLR     (SP)            ;CLEAR THE RUBOUT KEY
030624  122713  000025          7$:     CMPB    #25,(R3)        ;IS CHARACTER A CTRL U?
030630  001003                          BNE     10$             ;BR IF NO
030632  104401  033314                  TYPE    ,$CNTLU         ;TYPE A CONTROL "U"
030636  000726                          BR      1$              ;GO START OVER
030640  122713  000003          10$:    CMPB    #3,(R3)         ;IS CHARACTER A CTRL C ?
030644  001006                          BNE     8$              ;BR IF NOT
030646  012737  177777  001360          MOV     #-1,CFLAG       ;SET CNTRL C FLAG
030654  104401  031000                  TYPE    ,$CNTLC         ;ECHO IT
030660  000427                          BR      11$             ;EXIT
```

```
030662  122713  000012          8$:     CMPB    #12,(R3)          ;IS CHARACTER A "LF"?
030666  001011                          BNE     3$                ;BRANCH IF NO
030670  105013                          CLRB    (R3)              ;CLEAR THE CHARACTER
030672  104401  001201                  TYPE    ,$CRLF            ;TYPE A "CR" & "LF"
030676  104401  030766                  TYPE    ,$TTYIN           ;TYPE THE INPUT STRING
030702  000706                          BR      2$                ;GO PICKUP ANOTHER CHACTER
030704  104401  001200          4$:     TYPE    ,$QUES            ;TYPE A '?'
030710  000701                          BR      1$                ;CLEAR THE BUFFER AND LOOP
030712  111337  030764          3$:     MOVB    (R3),9$           ;ECHO THE CHARACTER
030716  104401  030764                  TYPE    ,9$
030722  122723  000015                  CMPB    #15,(R3)+         ;CHECK FOR RETURN
030726  001274                          BNE     2$                ;LOOP IF NOT RETURN
030730  105063  177777                  CLRB    -1(R3)            ;CLEAR RETURN (THE 15)
030734  104401  001202                  TYPE    ,$LF              ;TYPE A LINE FEED
030740  005726                  11$:    TST     (SP)+             ;CLEAN RUBOUT KEY FROM THE STACK
030742  012603                          MOV     (SP)+,R3          ;RESTORE R3
030744  011646                          MOV     (SP),-(SP)        ;ADJUST THE STACK AND PUT ADDRESS OF THE
030746  016666  000004  000002          MOV     4(SP),2(SP)       ;        FIRST ASCII CHARACTER ON IT
030754  012766  030766  000004          MOV     #$TTYIN,4(SP)
030762  000002                          RTI                       ;RETURN
030764     000                  9$:     .BYTE   0                 ;STORAGE FOR ASCII CHAR. TO TYPE
030765     000                          .BYTE   0                 ;TERMINATOR
030766  000012                  $TTYIN: .BLKB   10.               ;RESERVE 10 BYTES FOR TTY INPUT
031000  041536  005015     000  $CNTLC: .ASCIZ  /^C/<CR><LF>      ;CONTROL "C"

        031006                  .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

031006                          $ERROR:
031006  104407                          CKSWR                     ;;TEST FOR CHANGE IN SOFT-SWR
031010  010337  001342                  MOV     R3,ATTN           ;SAVE THE ATTENTION REGISTER CONTENTS
031014  010137  001216                  MOV     R1,DRIVE          ;DRIVE NUMBER
031020  032777  020000  150126          BIT     #SW13,@SWR        ;INHIBIT PRINTOUTS ?
031026  001002                          BNE     .+6               ;BR IF YES
031030  004737  023450                  JSR     PC,$TIME          ;TYPE THE TIME
031034  105237  001117          7$:     INCB    $ERFLG            ;;SET THE ERROR FLAG
031040  001775                          BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
031042  013777  001116  150106          MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
```

```
031050  032777  002000  150076          BIT     #BIT10,@$WR         ;;BELL ON ERROR?
031056  001402                          BEQ     1$                  ;;NO - SKIP
031060  104401  001174                  TYPE    ,$BELL              ;;RING BELL
031064  005237  001126          1$:     INC     $ERTTL              ;;COUNT THE NUMBER OF ERRORS
031070  011637  001132                  MOV     (SP),$ERRPC         ;;GET ADDRESS OF ERROR INSTRUCTION
031074  162737  000002  001132          SUB     #2,$ERRPC
031102  117737  150024  001130          MOVB    @$ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
031110  032777  020000  150036          BIT     #BIT13,@$WR         ;;SKIP TYPEOUT IF SET
031116  001004                          BNE     20$                 ;;SKIP TYPEOUTS
031120  004737  031172                  JSR     PC,$ERRTYP          ;;GO TO USER ERROR ROUTINE
031124  104401  001201                  TYPE    ,$CRLF
031130                          20$:
031130  122737  000001  001224          CMPB    #APTENV,$ENV        ;;RUNNING IN APT MODE
031136  001007                          BNE     2$                  ;;NO,SKIP APT ERROR REPORT
031140  113737  001130  031152          MOVB    $ITEMB,21$          ;;SET ITEM NUMBER AS ERROR NUMBER
031146  004737  031626                  JSR     PC,$ATY4            ;;REPORT FATAL ERROR TO APT
031152     000                  21$:    .BYTE   0
031153     000                          .BYTE   0
031154  000777                  22$:    BR      22$                 ;;APT ERROR LOOP
031156  005777  147772          2$:     TST     @$WR                ;;HALT ON ERROR
031162  100002                          BPL     3$                  ;;SKIP IF CONTINUE
031164  000000                          HALT                        ;;HALT ON ERROR!
031166  104407                          CKSWR                       ;;TEST FOR CHANGE IN SOFT-SWR
031170                          3$:
031170  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.

031172                          $ERRTYP:
031172  104401  001201                  TYPE    ,$CRLF              ;;"CARRIAGE RETURN" & "LINE FEED"
031176  010046                          MOV     R0,-(SP)            ;;SAVE R0
031200  005000                          CLR     R0                  ;;PICKUP THE ITEM INDEX
031202  153700  001130                  BISB    @#$ITEMB,R0
031206  001004                          BNE     1$                  ;;IF ITEM NUMBER IS ZERO, JUST
                                                                    ;;TYPE THE PC OF THE ERROR
031210  013746  001132                  MOV     $ERRPC,-(SP)        ;;SAVE $ERRPC FOR TYPEOUT
                                                                    ;;ERROR ADDRESS
031214  104402                          TYPOC                       ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
031216  000426                          BR      6$                  ;;GET OUT
031220  005300                  1$:     DEC     R0                  ;;ADJUST THE INDEX SO THAT IT WILL
031222  006300                          ASL     R0                  ;;      WORK FOR THE ERROR TABLE
031224  006300                          ASL     R0
031226  006300                          ASL     R0
031230  062700  003542                  ADD     #$ERRTB,R0          ;;FORM TABLE POINTER
031234  012037  031244                  MOV     (R0)+,2$            ;;PICKUP "ERROR MESSAGE" POINTER
031240  001404                          BEQ     3$                  ;;SKIP TYPEOUT IF NO POINTER
031242  104401                          TYPE                        ;;TYPE THE "ERROR MESSAGE"
031244  000000                  2$:     .WORD   0                   ;;"ERROR MESSAGE" POINTER GOES HERE
031246  104401  001201                  TYPE    ,$CRLF              ;;"CARRIAGE RETURN" & "LINE FEED"
031252  012037  031262          3$:     MOV     (R0)+,4$            ;;PICKUP "DATA HEADER" POINTER
031256  001404                          BEQ     5$                  ;;SKIP TYPEOUT IF 0
```

```
031260  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
031262  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
031264  104401  001201                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
031270  011000                  5$:     MOV     @(R0),R0        ;;PICKUP "DATA TABLE" POINTER
031272  001004                          BNE     7$              ;;GO TYPE THE DATA
031274  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
031276  104401  001201                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
031302  000207                          RTS     PC              ;;RETURN
031304                          7$:
031304  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
031306  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
031310  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
031312  001770                          BEQ     6$              ;;BR IF NO
031314  104401  031322                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
031320  000771                          BR      7$              ;;LOOP
031322  020040     000          8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
        031326                          .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
                                ;*

031326  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
031332  100002                          BPL     1$              ;;BR IF YES
031334  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
031336  000430                          BR      3$              ;;LEAVE
031340  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
031342  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
031346  122737  000001  001224          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
031354  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
031356  132737  000100  001225          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
031364  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
031366  010037  031376                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
031372  004737  031616                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
031376  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
031400  132737  000040  001225  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
031406  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
031410  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
031412  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
031414  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
031416  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
031420  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
031424  000002                          RTI                     ;;RETURN
```

```
031426  122716  000011          4$:     CMPB    #HT,(SP)          ;;BRANCH IF <HT>
031432  001430                          BEQ     8$
031434  122716  000200                  CMPB    #CRLF,(SP)        ;;BRANCH IF NOT <CRLF>
031440  001006                          BNE     5$
031442  005726                          TST     (SP)+             ;;POP  <CR><LF> EQUIV
031444  104401                          TYPE                      ;;TYPE A CR AND LF
031446  001201                          $CRLF
031450  105037  031604                  CLRB    $CHARCNT          ;;CLEAR CHARACTER COUNT
031454  000755                          BR      2$                ;;GET NEXT CHARACTER
031456  004737  031540          5$:     JSR     PC,$TYPEC         ;;GO TYPE THIS CHARACTER
031462  123726  001172          6$:     CMPB    $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
031466  001350                          BNE     2$                ;;IF NO GO GET NEXT CHAR.
031470  013746  001170                  MOV     $NULL,-(SP)       ;;GET # OF FILLER CHARS. NEEDED
                                                                  ;;AND THE NULL CHAR.
031474  105366  000001          7$:     DECB    1(SP)             ;;DOES A NULL NEED TO BE TYPED?
031500  002770                          BLT     6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
031502  004737  031540                  JSR     PC,$TYPEC         ;;GO TYPE A NULL
031506  105337  031604                  DECB    $CHARCNT          ;;DO NOT COUNT AS A COUNT
031512  000770                          BR      7$                ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

031514  112716  000040          8$:     MOVB    #'  ,(SP)         ;;REPLACE TAB WITH SPACE
031520  004737  031540          9$:     JSR     PC,$TYPEC         ;;TYPE A SPACE
031524  132737  000007  031604          BITB    #7,$CHARCNT       ;;BRANCH IF NOT AT
031532  001372                          BNE     9$                ;;TAB STOP
031534  005726                          TST     (SP)+             ;;POP SPACE OFF STACK
031536  000724                          BR      2$                ;;GET NEXT CHARACTER
031540  105777  147420          $TYPEC: TSTB    @$TPS             ;;WAIT UNTIL PRINTER IS READY
031544  100375                          BPL     $TYPEC
031546  116677  000002  147412          MOVB    2(SP),@$TPB       ;;LOAD CHAR TO BE TYPED INTO DATA REG.
031554  122766  000015  000002          CMPB    #CR,2(SP)         ;;IS CHARACTER A CARRIAGE RETURN?
031562  001003                          BNE     1$                ;;BRANCH IF NO
031564  105037  031604                  CLRB    $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
031570  000406                          BR      $TYPEX            ;;EXIT
031572  122766  000012  000002  1$:     CMPB    #LF,2(SP)         ;;IS CHARACTER A LINE FEED?
031600  001402                          BEQ     $TYPEX            ;;BRANCH IF YES
031602  105227                          INCB    (PC)+             ;;COUNT THE CHARACTER
031604  000000                  $CHARCNT:.WORD  0                 ;;CHARACTER COUNT STORAGE
031606  000207                  $TYPEX: RTS     PC


                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
031610  112737  000001  032054  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
031616  112737  000001  032052  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
031624  000403                          BR      $ATYC
031626  112737  000001  032054  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
031634                          $ATYC:
031634  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
031636  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
031640  105737  032052                  TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
031644  001450                          BEQ     5$                ;;IF NOT:  BR
031646  122737  000001  001224          CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
031654  001031                          BNE     3$                ;;IF NOT:  BR
```

```
031656  132737  000100  001225          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
031664  001425                          BEQ     3$              ;;IF NOT:  BR
031666  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
031672  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
031700  005737  001204          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
031704  001375                          BNE     1$              ;;IF NOT:  WAIT
031706  010037  001220                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
031712  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
031714  001376                          BNE     2$
031716  163700  001220                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
031722  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
031724  010037  001222                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
031730  012737  000004  001204          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
031736  000413                          BR      5$
031740  017637  000004  031764  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
031746  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
031754  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
031760  004737  031326                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
031764  000000                  4$:     .WORD   0
031766                          5$:
031766  105737  032054          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
031772  001416                          BEQ     12$             ;;IF NOT:  BR
031774  005737  001224                  TST     $ENV            ;;RUNNING UNDER APT?
032000  001413                          BEQ     12$             ;;IF NOT:  BR
032002  005737  001204          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
032006  001375                          BNE     11$             ;;IF NOT:  WAIT
032010  017637  000004  001206          MOV     @4(SP),$FATAL   ;;GET ERROR #
032016  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
032024  005237  001204                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
032030  105037  032054          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
032034  105037  032053                  CLRB    $LFLG           ;;CLEAR LOG FLAG
032040  105037  032052                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
032044  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
032046  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
032050  000207                          RTS     PC              ;;RETURN
032052     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
032053     000                  $LFLG:  .BYTE   0               ;;LOG FLAG
032054     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
            032056                          .EVEN
            000200              APTSIZE=200
            000001              APTENV=001
            000100              APTSPOOL=100
            000040              APTCSUP=040

                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*****************************************************************
                                ;POWER DOWN ROUTINE
032056  012737  032222  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
032064  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
032072  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
032074  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
032076  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
032100  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
032102  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
032104  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
```

```
032106  017746  147042                  MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
032112  010637  032226                  MOV     SP,$SAVR6         ;;SAVE SP
032116  012737  032130  000024          MOV     #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
032124  000000                          HALT
032126  000776                          BR      .-2               ;;HANG UP

                                ;;*************************************************************
                                ;POWER UP ROUTINE
032130  012737  032222  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
032136  013706  032226                  MOV     $SAVR6,SP         ;;GET SP
032142  005037  032226                  CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
032146  005237  032226          1$:     INC     $SAVR6            ;;WAIT FOR THE INC
032152  001375                          BNE     1$                ;;OF  WORD
032154  012677  146774                  MOV     (SP)+,@SWR        ;;POP STACK INTO @SWR
032160  012605                          MOV     (SP)+,R5          ;;POP STACK INTO R5
032162  012604                          MOV     (SP)+,R4          ;;POP STACK INTO R4
032164  012603                          MOV     (SP)+,R3          ;;POP STACK INTO R3
032166  012602                          MOV     (SP)+,R2          ;;POP STACK INTO R2
032170  012601                          MOV     (SP)+,R1          ;;POP STACK INTO R1
032172  012600                          MOV     (SP)+,R0          ;;POP STACK INTO R0
032174  012737  032056  000024          MOV     #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
032202  012737  000340  000026          MOV     #340,@#PWRVEC+2   ;;PRIO:7
032210  104401                          TYPE                      ;REPORT THE POWER FAILURE
032212  032230                  $PWRMG: .WORD   $POWER            ;;POWER FAIL MESSAGE POINTER
032214  012716                          MOV     (PC)+,(SP)        ;;RESTART AT START2
032216  003650                  $PWRAD: .WORD   START2            ;;RESTART ADDRESS
032220  000002                          RTI
032222  000000                  $ILLUP: HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
032224  000776                          BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
032226  000000                  $SAVR6: 0                         ;;PUT THE SP HERE
032230  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
032236  000122
                                        .EVEN

                                .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

032240  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
032244  116637  000001  032463          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
032252  112637  032465                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
032256  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
032262  000406                          BR      $TYPON
032264  112737  000001  032463  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
032272  112737  000006  032465          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
032300  112737  000005  032462  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
032306  010346                          MOV     R3,-(SP)        ;;SAVE R3
032310  010446                          MOV     R4,-(SP)        ;;SAVE R4
032312  010546                          MOV     R5,-(SP)        ;;SAVE R5
032314  113704  032465                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
032320  005404                          NEG     R4
032322  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
032326  110437  032464                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
032332  113704  032463                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
032336  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
032342  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
032344  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
032346  000404                          BR      3$              ;;GO DO MSB
032350  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
032352  006105                          ROL     R5
032354  006105                          ROL     R5
032356  010503                          MOV     R5,R3
032360  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
032362  105337  032464                  DECB    $OMODE          ;;TYPE THIS DIGIT?
032366  100016                          BPL     7$              ;;BR IF NO
032370  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
032374  001002                          BNE     4$              ;;TEST FOR 0
032376  005704                          TST     R4              ;;SUPPRESS THIS 0?
032400  001403                          BEQ     5$              ;;BR IF YES
032402  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
032404  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
032410  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
032414  110337  032460                  MOVB    R3,8$           ;;SAVE FOR TYPING
032420  104401  032460                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
032424  105337  032462          7$:     DECB    $OCNT           ;;COUNT BY 1
032430  003347                          BGT     2$              ;;BR IF MORE TO DO
032432  002402                          BLT     6$              ;;BR IF DONE
032434  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
032436  000744                          BR      2$              ;;GO DO THE LAST DIGIT
032440  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
032442  012604                          MOV     (SP)+,R4        ;;RESTORE R4
032444  012603                          MOV     (SP)+,R3        ;;RESTORE R3
032446  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
032454  012616                          MOV     (SP)+,(SP)
032456  000002                          RTI                     ;;RETURN
032460     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
032461     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
032462     000          $OCNT:          .BYTE   0               ;;OCTAL DIGIT COUNTER
032463     000          $OFILL:         .BYTE   0               ;;ZERO FILL SWITCH
032464  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

032466                          $TYPDS:
032466  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
032470  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
032472  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
032474  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
032476  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
032500  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
032504  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
032510  100004                          BPL     1$              ;;BR IF INPUT IS POS.
032512  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
032514  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
032522  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
032524  012703  032702                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
032530  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
032534  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
032536  016001  032672                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
032542  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
032544  002402                          BLT     4$              ;;BR IF DONE
032546  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
032550  000774                          BR      3$
032552  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
032554  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
032556  001002                          BNE     5$              ;;FALL THROUGH IF 0
032560  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
032562  100407                          BMI     7$              ;;BR IF YES
032564  106316                  5$:     ASLB    (SP)            ;;MSD?
032566  103003                          BCC     6$              ;;BR IF NO
032570  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
032576  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
032602  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
032606  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
032610  005720                          TST     (R0)+           ;;JUST INCREMENTING
032612  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
032616  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
032620  003002                          BGT     8$              ;;GO TO EXIT
032622  010502                          MOV     R5,R2           ;;GET THE LSD
032624  000764                          BR      6$              ;;GO CHANGE TO ASCII
032626  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
032630  100003                          BPL     9$              ;;BR IF NO
032632  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
032640  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
032642  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
032644  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
032646  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
032650  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
032652  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
032654  104401  032702                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
032660  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
032666  012616                          MOV     (SP)+,(SP)
032670  000002                          RTI                     ;;RETURN TO USER
032672  023420                  $DTBL:  10000.
032674  001750                          1000.
032676  000144                          100.
032700  000012                          10.
032702  000004                  $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.
032712  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
032720  001074                          BNE     15$             ;;BRANCH IF NO
032722  105777  146232                  TSTB    @$TKS           ;;CHAR THERE?
032726  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
032730  117746  146226                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
032734  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
032740  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
032744  001062                          BNE     15$             ;;NO, RETURN TO USER
032746  123727  001150  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
032754  001456                          BEQ     15$             ;;BRANCH IF YES

032756  104401  033321                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (↑G)
032762  104401  033326          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
032766  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
032772  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032774  104401  033337                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
033000  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
033002  005046                          CLR     -(SP)           ;;THE NEW SWR
033004  105777  146150          7$:     TSTB    @$TKS           ;;CHAR THERE?
033010  100375                          BPL     7$              ;;IF NOT TRY AGAIN

033012  117746  146144                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
033016  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


033022  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
033026  001005                          BNE     10$             ;;BRANCH IF NOT
033030  104401  033314                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
033034  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
033040  000757                          BR      19$             ;;LET'S TRY IT AGAIN


033042  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
```

```
033046  001022                          BNE     16$             ;;BRANCH IF NO
033050  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
033054  001403                          BEQ     11$             ;;BRANCH IF YES
033056  016677  000002  146070          MOV     2(SP),@SWR      ;;SAVE NEW SWR
033064  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
033070  104401  001201          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
033074  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
033102  001003                          BNE     15$             ;;BRANCH IF NOT
033104  012777  000100  146046          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
033112  000002                  15$:    RTI                     ;;RETURN
033114  004737  031540          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
033120  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
033124  002420                          BLT     18$             ;;BRANCH IF YES
033126  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
033132  003015                          BGT     18$             ;;BRANCH IF YES
033134  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
033140  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
033144  001403                          BEQ     17$             ;;BRANCH IF YES
033146  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
033150  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
033152  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
033154  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
033160  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
033164  000707                          BR      7$              ;;GET THE NEXT ONE
033166  104401  001200          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
033172  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
                                ;
033174  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
033176  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
033204  105777  145750          1$:     TSTB    @$TKS           ;;WAIT FOR
033210  100375                          BPL     1$              ;;A CHARACTER
033212  117766  145744  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
033220  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
033226  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
033234  001013                          BNE     3$              ;;BRANCH IF NO
033236  105777  145716          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
033242  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
033244  117746  145712                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
033250  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
033254  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
033260  001366                          BNE     2$              ;;IF NOT DISCARD IT
033262  000750                          BR      1$              ;;YES, RESUME
033264  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
033272  002407                          BLT     4$              ;;BRANCH IF YES
033274  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
033302  003003                          BGT     4$              ;;BRANCH IF YES
```

```
033304  042766  000040  000004          BIC     #40,4(SP)        ;;MAKE IT UPPER CASE
033312  000002                  4$:     RTI                      ;;GO BACK TO USER
033314  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
033321     136  006507  000012  $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
033326  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
033334  020075     000
033337     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
033344  036440  000040

                                .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

033350                          $RAND:
033350  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
033352  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
033354  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
033356  013700  033450                  MOV     $LONUM,R0        ;;SET R0 WITH LOW
033362  013701  033446                  MOV     $HINUM,R1        ;;SET R1 WITH HIGH
033366  012702  177771                  MOV     #-7,R2           ;;SET SHIFT COUNT
033372  006300                  1$:     ASL     R0               ;;SHIFT R0 LEFT AND
033374  006101                          ROL     R1               ;;ROTATE CARRY INTO R1 AND
033376  005202                          INC     R2               ;;CHECK FOR DONE
033400  001374                          BNE     1$               ;;CONTINUE SHIFT LOOP
033402  063700  033450                  ADD     $LONUM,R0        ;;ADD NUMBER TO MAKE X 129
033406  005501                          ADC     R1               ;;PROPOGATE CARRY
033410  063701  033446                  ADD     $HINUM,R1        ;;ADD NUMBER TO MAKE X 129
033414  062700  001057                  ADD     #1057,R0         ;;ADD LOW CONSTANT
033420  005501                          ADC     R1               ;;PROPOGATE CARRY
033422  062701  047401                  ADD     #47401,R1        ;;ADD HIGH CONSTANT
033426  010037  033450                  MOV     R0,$LONUM        ;;SAVE R0
033432  010137  033446                  MOV     R1,$HINUM        ;;SAVE R1
033436  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
033440  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
033442  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
033444  000207                          RTS     PC               ;;RETURN
033446  176543                  $HINUM: .WORD   176543
033450  123456                  $LONUM: .WORD   123456

                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                                ;;*****************************************************************
                                ;*SAVE R0-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---R0

033452                                  $SAVREG:
033452  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033454  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033456  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
033460  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
033462  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
033464  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
033466  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
033472  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
033476  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
033502  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
033506  000002                                  RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
033510                                  $RESREG:
033510  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
033514  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
033520  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
033524  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
033530  012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
033532  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
033534  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
033536  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
033540  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
033542  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
033544  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

                                        $DB2D:  SAVREG                  ;;SAVE REGISTERS
033546  104412                                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
033550  016602  000002                          MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
033554  012700  033726                          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
033560  010066  000002                          MOV     (R2),R1         ;;PICKUP THE BINARY NUMBER
033564  012201                                  MOV     (R2)+,R2
033566  012202                                  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
033570  012737  000012  033644
```

```
033576  012704  033656                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
033602  012705  033660                  MOV     #$TNPWR+2,R5
033606  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
033610  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
033612  005602                          SBC     R2
033614  161502                          SUB     (R5),R2
033616  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
033620  005203                          INC     R3              ;;ADD 1 TO PARTIAL
033622  000772                          BR      2$              ;;LOOP
033624  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
033626  005502                          ADC     R2
033630  062402                          ADD     (R4)+,R2
033632  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
033634  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
033640  110320                          MOVB    R3,(R0)+        ;;SAVE IT
033642  005327                          DEC     (PC)+           ;;DONE?
033644  000000                  4$:     .WORD   0
033646  001357                          BNE     1$              ;;BR IF NO
033650  105020                          CLRB    (R0)+           ;;TERMINATOR
033652  104413                          RESREG                  ;;RESTORE REGISTERS
033654  000207                          RTS     PC              ;;RETURN
033656  145000                  $TNPWR: 145000                  ;;1.0E09
033660  035632                          35632
033662  160400                          160400                  ;;1.0E08
033664  002765                          2765
033666  113200                          113200                  ;;1.0E07
033670  000230                          230
033672  041100                          041100                  ;;1.0E06
033674  000017                          17
033676  103240                          103240                  ;;1.0E05
033700  000001                          1
033702  023420                          23420                   ;;1.0E04
033704  000000                          0
033706  001750                          1750                    ;;1.0E03
033710  000000                          0
033712  000144                          144                     ;;1.0E02
033714  000000                          0
033716  000012                          12                      ;;1.0E01
033720  000000                          0
033722  000001                          1                       ;;1.0E00
033724  000000                          0
033726  000014                  $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,@#$DB20      ;;CALL THE ROUTINE
                                ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

033742  104412                  $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
033744  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
```

```
033750  012705  034061                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
033754  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
033760  012703  177770                  MOV     #↑C7,R3         ;;MASK
033764  012100                          MOV     (R1)+,R0        ;;LOWER WORD
033766  012101                          MOV     (R1)+,R1        ;;HIGH WORD
033770  005002                          CLR     R2              ;;TERMINATOR
033772  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
033774  010002                          MOV     R0,R2           ;;GET THIS DIGIT
033776  005304                          DEC     R4              ;;COUNT THIS CHARACTER
034000  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
034002  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
034004  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
034006  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
034012  104413                          RESREG                  ;;RESTORE ALL REGISTERS
034014  000207                          RTS     PC              ;;RETURN TO USER
034016  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
034020  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
034022  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
034024  006001                          ROR     R1
034026  006000                          ROR     R0
034030  006001                          ROR     R1
034032  006000                          ROR     R0
034034  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
034036  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
034042  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
034044  000016                  $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.

034062  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
034064  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
034070  005740                          TST     -(R0)           ;;BACKUP BY 2
034072  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
034074  006300                          ASL     R0              ;;POSITION FOR INDEXING
034076  016000  034116                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
034102  000200                          RTS     R0              ;;GO TO ROUTINE


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

034104  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
034106  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
034114  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
```

```
                                        ;       -------
034116  034104                          $TRPAD: .WORD   $TRAP2
034120  031326                                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
034122  032264                                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
034124  032240                                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
034126  032300                                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
034130  032466                                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

034132  032762                                  $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

034134  032712                                  $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
034136  033174                                  $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
034140  030510                                  $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
034142  033452                                  $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
034144  033510                                  $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
034146  027734                                  $DSPLY  ;;CALL=DISPLY   TRAP+14(104414) ROUTINE TO TYPE ERROR MESSAGES
        000032                          $TERM=.-$TRPAD

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


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

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

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

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

034150  000000  000000  000000          RMERRS: .WORD   0,0,0,0
034156  000000

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

034160  000                             DRVACT: .BYTE   0               ;DRIVE 0
034161  000                                     .BYTE   0               ;DRIVE 1
034162  000                                     .BYTE   0               ;DRIVE 2
034163  000                                     .BYTE   0               ;DRIVE 3
034164  000                                     .BYTE   0               ;DRIVE 4
034165  000                                     .BYTE   0               ;DRIVE 5
034166  000                                     .BYTE   0               ;DRIVE 6
034167  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

034170  000                     DRVSTA: .BYTE   0               ;DRIVE 0
034171  000                             .BYTE   0               ;DRIVE 1
034172  000                             .BYTE   0               ;DRIVE 2
034173  000                             .BYTE   0               ;DRIVE 3
034174  000                             .BYTE   0               ;DRIVE 4
034175  000                             .BYTE   0               ;DRIVE 5
034176  000                             .BYTE   0               ;DRIVE 6
034177  000                             .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
                                        ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
                                        ;DRVTYP=4 IF DRIVE IS RM03
                                        ;DRVTYP=-1 IF NOT RM03

034200  000                     DRVTYP: .BYTE   0               ;DRIVE 0
034201  000                             .BYTE   0               ;DRIVE 1
034202  000                             .BYTE   0               ;DRIVE 2
034203  000                             .BYTE   0               ;DRIVE 3
034204  000                             .BYTE   0               ;DRIVE 4
034205  000                             .BYTE   0               ;DRIVE 5
034206  000                             .BYTE   0               ;DRIVE 6
034207  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

034210  000                     DPINT:  .BYTE   0               ;DRIVE 0
034211  000                             .BYTE   0               ;DRIVE 1
034212  000                             .BYTE   0               ;DRIVE 2
034213  000                             .BYTE   0               ;DRIVE 3
034214  000                             .BYTE   0               ;DRIVE 4
034215  000                             .BYTE   0               ;DRIVE 5
034216  000                             .BYTE   0               ;DRIVE 6
034217  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

034220  000                     DPRQS:  .BYTE   0               ;DRIVE 0
034221  000                             .BYTE   0               ;DRIVE 1
034222  000                             .BYTE   0               ;DRIVE 2
034223  000                             .BYTE   0               ;DRIVE 3
034224  000                             .BYTE   0               ;DRIVE 4
034225  000                             .BYTE   0               ;DRIVE 5
034226  000                             .BYTE   0               ;DRIVE 6
034227  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.
```

```
034230  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.

034232  000000          SRCHWT: .WORD   0

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

034234     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

034235     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

034236     000          ULDFLG: .BYTE   0               ;DRIVE 0
034237     000                  .BYTE   0               ;DRIVE 1
034240     000                  .BYTE   0               ;DRIVE 2
034241     000                  .BYTE   0               ;DRIVE 3
034242     000                  .BYTE   0               ;DRIVE 4
034243     000                  .BYTE   0               ;DRIVE 5
034244     000                  .BYTE   0               ;DRIVE 6
034245     000                  .BYTE   0               ;DRIVE 7

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

034246     000          LACNT:  .BYTE   0               ;DRIVE 0
034247     000                  .BYTE   0               ;DRIVE 1
034250     000                  .BYTE   0               ;DRIVE 2
034251     000                  .BYTE   0               ;DRIVE 3
034252     000                  .BYTE   0               ;DRIVE 4
034253     000                  .BYTE   0               ;DRIVE 5
034254     000                  .BYTE   0               ;DRIVE 6
034255     000                  .BYTE   0               ;DRIVE 7

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

034256  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

034260  177777                          SEEKFG: .WORD   -1

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

034262  177777                          TIMER:  .WORD   -1      ;DRIVE 0
034264  177777                                  .WORD   -1      ;DRIVE 1
034266  177777                                  .WORD   -1      ;DRIVE 2
034270  177777                                  .WORD   -1      ;DRIVE 3
034272  177777                                  .WORD   -1      ;DRIVE 4
034274  177777                                  .WORD   -1      ;DRIVE 5
034276  177777                                  .WORD   -1      ;DRIVE 6
034300  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

034302  177777                          DTUW:   .WORD   -1

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

034304     001                          ATABIT: .BYTE   1       ;DRIVE 0
034305     002                                  .BYTE   2       ;DRIVE 1
034306     004                                  .BYTE   4       ;DRIVE 2
034307     010                                  .BYTE   10      ;DRIVE 3
034310     020                                  .BYTE   20      ;DRIVE 4
034311     040                                  .BYTE   40      ;DRIVE 5
034312     100                                  .BYTE   100     ;DRIVE 6
034313     200                                  .BYTE   200     ;DRIVE 7

                                        ;FSRM03 TO RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
                                        ;CALLING IT FATAL (MCPEMX=1 WORD)

034314  000003                          MCPEMX: .WORD   3

                                        ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH70/RM03),
                                        ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).

034316  176700                          RMADR:  .WORD   176700
034320  000254  000240                  RMVEC:  .WORD   254,5*32.

                                        ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)

034324  000004                          MXLACT: .WORD   4
                                        ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)

034326  001000                          MXDLTA: .WORD   8.*64.
```

```
                                        ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

034330  000200                          MNDLTA: .WORD   2*64.
                                        ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)

034332  000005                          MXWNDW: .WORD   5

                                        ;DEFINITIONS OF THE RH70/RM03 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)

                                        ;RH70/RM03 DRIVER INITIALIZATION CODE
                                                ;THIS ROUTINE WILL DETERMINE WHICH RM03 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
                                        ;
034334  104412                          RMINIT: SAVREG                  ;SAVE R0 - R5
034336  013746  177776                          MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
034342  012737  000240  177776                  MOV     #<5*32.>,@#PS   ;CHANGE THE PRIORITY TO 5
034350  004737  042230                          JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
034354  012701  034150                          MOV     #RMERRS,R1      ;FIRST ADDRESS TO BE CLEARED
034360  012702  034260                          MOV     #SEEKFG,R2      ;LAST ADDRESS TO BE CLEARED
034364  005021                          1$:     CLR     (R1)+           ;CLEAR
034366  020102                                  CMP     R1,R2           ;ARE WE DONE?
034370  103775                                  BLO     1$              ;BRANCH IF NO
034372  012702  034302                          MOV     #DTUW,R2        ;LAST ADDRESS
034376  012721  177777                  2$:     MOV     #-1,(R1)+       ;INITIALIZE
034402  020102                                  CMP     R1,R2           ;DONE?
034404  101774                                  BLOS    2$              ;LOOP IF NO
```

```
034406  005037  034170                  CLR     DRVSTA          ;SET ALL DRIVES TO OFFLINE
034412  005037  034172                  CLR     DRVSTA+2
034416  005037  034174                  CLR     DRVSTA+4
034422  005037  034176                  CLR     DRVSTA+6
034426  013703  034320                  MOV     RMVEC,R3        ;SETUP THE RH70/RM03 VECTOR
034432  012723  037122                  MOV     #ISR,(R3)+
034436  013713  034322                  MOV     RMVEC+2,(R3)
034442  013704  034316                  MOV     RMADR,R4        ;FIRST ADDRESS OF RH70/RM03
034446  012764  000040  000010          MOV     #BIT0$,RMCS2(R4)  ;MASSBUS INIT
034454  005001                          CLR     R1              ;START WITH DRIVE 0
034456  004037  034546          3$:     JSR     R0,DRVINT       ;INIT THE DRIVE
034462  000401                          BR      4$              ;'DVA' NOT SET OR PARITY ERROR
034464  000402                          BR      5$              ;NORMAL RETURN
034466  105061  034170          4$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
034472  005201                  5$:     INC     R1              ;GO TO NEXT DRIVE
034474  042701  177770                  BIC     #^C7,R1         ;MASK OUT UNUSED BITS
034500  001366                          BNE     3$              ;BR IF MORE DRIVES TO GO
034502  012701  000007                  MOV     #7,R1           ;START WITH DRIVE 7
034506  005037  177776                  CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
034512  105761  034210          6$:     TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
034516  001405                          BEQ     8$              ;BR NOT WAITING
034520  004737  041664                  JSR     PC,SET.IE       ;SET INTERRUPT
034524  105761  034210          7$:     TSTB    DPINT(R1)       ;DRIVE SWITCHED PORTS ?
034530  001375                          BNE     7$              ;BR IF NOT
034532  005301                  8$:     DEC     R1              ;GO TO THE NEXT DRIVE
034534  100366                          BPL     6$              ;CHECK NEXT DRIVE
034536  012637  177776                  MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
034542  104413                          RESREG                  ;RESTORE R0 - R5
034544  000207                          RTS     PC              ;BYE-BYE

                                ;DRIVE INITILIZATION ROUTINE
                                        ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
                                        ;AN RM03. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
                                        ;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 RH70/RM03 (RMCS1)
                                ;       JSR     R0,DRVINT       ;CALLED BY A JSR
                                ;       RETURN1                 ;ERROR OCCURRED (PARITY)
                                ;       RETURN2                 ;NORMAL RETURN
                                ;

034546  010546                  DRVINT: MOV     R5,-(SP)        ;SAVE R5
034550  105061  034170          DULP:   CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
034554  105061  034200                  CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
034560  105061  034236                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
034564  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT A DRIVE
034570  112764  000111  000000          MOVB    #111,RMCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
034576  032764  010000  000010          BIT     #BIT12,RMCS2(R4) ;NONEXISTENT DRIVE?
034604  001403                          BEQ     1$              ;NO---BRANCH
034606  004737  041664                  JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
034612  000476                          BR      6$              ;LEAVE THIS ROUTINE
034614  105061  034170          1$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
034620  032764  004000  000000          BIT     #BIT11,RMCS1(R4)  ;SEE IF DRIVE AVAILABLE
```

```
034626  001750                          BEQ     DULP            ;BR IF DRIVE NOT AVAILABLE
034630  004037  041204                  JSR     R0,RD.RM        ;READ THE DRIVE TYPE REG.
034634  000026                          RMDT
034636  035034                          8$                      ;ERROR RETURN ADDRESS
034640  012605                          MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
034642  112761  000004  034200          MOVB    #4,DRVTYP(R1)   ;SET RM03 INDICATOR
034650  022705  020024                  CMP     #20024,R5       ;SINGLE PORT RM03 ?
034654  001407                          BEQ     2$              ;BR IF YES
034656  022705  024024                  CMP     #24024,R5       ;DUAL PORT RM03 ?
034662  001404                          BEQ     2$              ;BR IF YES
034664  112761  177777  034200          MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
034672  000446                          BR      6$              ;EXIT
034674  012746  000121          2$:     MOV     #121,-(SP)      ;DO A "READ-IN PRESET"
034700  004037  041360                  JSR     R0,WRT.RM
034704  000000                          RMCS1
034706  035034                          8$
034710  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT22=1
034714  004037  041360                  JSR     R0,WRT.RM
034720  000032                          RMOF
034722  035034                          8$
034724  004037  041204                  JSR     R0,RD.RM        ;READ RMDS
034730  000012                          RMDS
034732  035034                          8$
034734  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
034736  100015                          BPL     4$              ;BRANCH IF ATA=0
034740  116164  034304  000016          MOVB    ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
034746  004037  041204                  JSR     R0,RD.RM        ;FIND OUT WHY ATA=1
034752  000014                          RMER1
034754  035034                          8$
034756  006126                          ROL     (SP)+           ;IS IT UNSAFE?
034760  100004                          BPL     4$              ;BR IF NOT
034762  112761  177777  034170          MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
034770  000407                          BR      6$              ;EXIT
034772  005105                  4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
034774  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
035000  001003                          BNE     6$              ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
035002  112761  000001  034170          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
035010  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
035012  000410                          BR      8$              ;EXIT
035014  006301                  7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
035016  012761  003720  034262          MOV     #2000.,TIMER(R1)  ;START 2 SEC TIMER
035024  006201                          ASR     R1              ;RESTORE R1
035026  112761  177777  034210          MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INIDICATOR
035034  012605                  8$:     MOV     (SP)+,R5        ;RESTORE R5
035036  000200                          RTS     R0              ;EXIT

                                ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
                                ;
                                ;CALL
                                ;
                                ;       JSR     R0,@#RM03       ;CALL THE RM03 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
```

```
035040  013746  177776          RM03:   MOV     @#PS,-(SP)          ;SAVE THE CALLING STATUS
035044  013737  034322  177776          MOV     RMVEC+2,@#PS        ;DON'T ALLOW ANY RM03 INTERRUPTS
035052  112737  000001  034234          MOVB    #1,ACTDRV           ;SET "ACTIVE DRIVER" FLAG
035060  104412                          SAVREG                      ;SAVE R0 - R5
035062  011002                          MOV     (R0),R2             ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
035064  005062  000016                  CLR     16(R2)              ;CLEAR THE STATUS/ERROR INDICATOR
035070  111201                          MOVB    (R2),R1             ;PICKUP THE DRIVE NUMBER
035072  013704  034316                  MOV     RMADR,R4            ;UNIBUS ADDRESS OF RMCS1
035076  105761  034170                  TSTB    DRVSTA(R1)          ;CHECK DRIVES STATUS
035102  003014                          BGT     1$                  ;BRANCH IF ONLINE
035104  105761  034236                  TSTB    ULDFLG(R1)          ;UNLOAD COMMAND IN QUEUE?
035110  001036                          BNE     3$                  ;BRANCH IF YES
035112  105761  034210                  TSTB    DPINT(R1)           ;TRYING TO INIT THE DRIVE
035116  001042                          BNE     5$                  ;BR IF YES
035120  004037  034546                  JSR     R0,DRVINT           ;GO INIT. THE DRIVE
035124  000434                          BR      4$                  ;ERROR RETURN
035126  105761  034170                  TSTB    DRVSTA(R1)          ;IS DRIVE STATUS ONLINE?
035132  003445                          BLE     6$                  ;BR IF NOT
035134  105761  034220          1$:     TSTB    DPRQS(R1)           ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
035140  001031                          BNE     5$                  ;BR IF YES
035142  010164  000010                  MOV     R1,RMCS2(R4)        ;SELECT THE DRIVE
035146  004037  042326                  JSR     R0,DRVQUE           ;PUT THIS REQUEST IN QUEUE
035152  000460                          BR      9$                  ;QUEUE IS FULL
035154  122762  000103  000002          CMPB    #103,2(R2)          ;IS THIS REQ. FOR AN UNLOAD?
035162  001003                          BNE     2$                  ;BR IF NO
035164  112761  177777  034236          MOVB    #-1,ULDFLG(R1)      ;SET THE "UNLOAD IN QUEUE" FLAG
035172  105761  034160          2$:     TSTB    DRVACT(R1)          ;IS THIS DRIVE ACTIVE?
035176  001043                          BNE     8$                  ;BR IF YES
035200  004737  035332                  JSR     PC,OPT              ;CALL THE OPTIMIZER
035204  000440                          BR      8$
035206  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
035214  000434                          BR      8$                  ;EXIT
035216  004737  036412          4$:     JSR     PC,CI7              ;GO HANDLE THE PARITY ERROR
035222  000431                          BR      8$
035224  004037  042326          5$:     JSR     R0,DRVQUE           ;PUT REQUEST IN QUEUE
035230  000431                          BR      9$                  ;QUEUE IS FULL
035232  032714  000100                  BIT     #BIT06,(R4)         ;IE BIT SET ?
035236  001023                          BNE     8$                  ;YES
035240  004737  041664                  JSR     PC,SET.IE           ;SET THE INTERRUPT
035244  000420                          BR      8$                  ;RETURN
035246  105761  034170          6$:     TSTB    DRVSTA(R1)          ;SEE IF DRIVE OFFLINE OR UNSAFE
035252  002412                          BLT     7$                  ;BR IF UNSAFE
035254  012762  140000  000016          MOV     #BIT15!BIT14,16(R2) ;SET OFFLINE ERROR INDICATOR
035262  105761  034200                  TSTB    DRVTYP(R1)          ;SEE IF OFFLINE OR NONEXISTENT
035266  001007                          BNE     8$                  ;BR IF OFFLINE
035270  012762  100002  000016          MOV     #BIT15!BIT01,16(R2) ;REPORT DRIVE NONEXISTENT
035276  000403                          BR      8$                  ;GO TO EXIT
035300  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2) ;DRIVE IS UNSAFE
035306  104413                  8$:     RESREG                      ;RESTORE R0 - R5
035310  005720                          TST     (R0)+               ;SETUP FOR NORMAL RETURN
035312  000401                          BR      10$                 ;FINISH UP, THEN EXIT
035314  104413                  9$:     RESREG                      ;RESTORE R0 - R5
035316  005720                  10$:    TST     (R0)+               ;CORRECT THE RETURN ADDRESS
035320  105037  034234                  CLRB    ACTDRV              ;CLEAR "ACTIVE DRIVER" FLAG
035324  012637  177776                  MOV     (SP)+,@#PS          ;RETURN "PS" TO USER LEVEL
035330  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
                                        ;
035332  104412                          OPT:    SAVREG                  ;SAVE R0 - R5
035334  013746  177776                          MOV     @#PS,-(SP)      ;SAVE PROC. STATUS
035340  146137  034304  034232                  BICB    ATABIT(R1),SRCHWT       ;CLEAR LA SEACH FLAG
035346  105061  034220                          CLRB    DPRQS(R1)       ;RESET THE PORT REQ FLAG ****
035352  004737  042402                          JSR     PC,GETREQ       ;GET "DPB" POINTER OF REQUEST
035356  005702                                  TST     R2              ;IS THERE A REQUEST IN QUEUE?
035360  001472                                  BEQ     7$              ;NO--BRANCH TO EXIT
035362  010164  000010                          MOV     R1,RMCS2(R4)    ;LOAD THE DRIVE ADDRESS *******
035366  012764  000111  000000                  MOV     #111,RMCS1(R4)  ;CLEAR THE DRIVE
035374  032764  004000  000000                  BIT     #BIT11,RMCS1(R4)        ;DVA SET ?
035402  001446                                  BEQ     5$              ;TO PROT REQUEST ,IF NOT
035404  105761  034170                  10$:    TSTB    DRVSTA(R1)      ;IS DRIVE ONLINE?
035410  003014                                  BGT     1$              ;YES--BRANCH
035412  004737  042424                          JSR     PC,POPQUE       ;NO--REMOVE REQUEST FROM QUEUE
035416  012762  140000  000016                  MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE STATUS/ERROR INDICATOR
035424  105761  034170                          TSTB    DRVSTA(R1)      ;IS DRIVE UNSAFE ?
035430  100053                                  BPL     8$              ;BR TO EXIT IF NOT
035432  012762  110000  000016                  MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
035440  000447                                  BR      8$              ;BRANCH TO EXIT
035442                                  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
035442  122762  000150  000002                  CMPB    #150,2(R2)      ;IS THE REQUEST FOR I/O?
035450  002403                                  BLT     2$              ;YES--BRANCH
035452  004737  035776                          JSR     PC,CI4          ;CALL THE COMMAND INITIATOR
035456  000440                                  BR      8$              ;BRANCH TO EXIT
035460  005737  034302                  2$:     TST     DTUW            ;DATA TRANSFER UNDERWAY?
035464  002012                                  BGE     4$              ;YES--GO START A SEARCH
035466  005737  034260                          TST     SEEKFG          ;DO IMPLIED SEEKS?
035472  100404                                  BMI     3$              ;YES---BRANCH
035474  004037  036746                          JSR     R0,LA           ;NO--DO LOOK AHEAD
035500  000427                                  BR      8$              ;RETURN HERE ON A PARITY ERROR
035502  000403                                  BR      4$              ;GO START A SEARCH
035504  004737  035570                  3$:     JSR     PC,CI1          ;START A DATA TRANSFER
035510  000423                                  BR      8$
035512  004737  035676                  4$:     JSR     PC,CI3          ;START A SEARCH
035516  000420                                  BR      8$              ;GO TO THE EXIT
035520  112761  177777  034220          5$:     MOVB    #-1,DPRQS(R1)   ;SET PORT REQUEST INDICATOR
035526  010103                                  MOV     R1,R3           ;SET UP TO ADDRESS WORDS
035530  006303                                  ASL     R3              ;CONVERT TO WORD INDEX
035532  012763  023420  034262                  MOV     #10000.,TIMER(R3)   ;START 10 SEC TIMER
035540  000402                                  BR      7$              ;EXIT
035542  004737  036412                  6$:     JSR     PC,CI7          ;PROCESS THE PARITY ERROR
035546  032714  000100                  7$:     BIT     #BIT06,(R4)     ;SEE IF 'IE' ALREADY SET
035552  001002                                  BNE     8$              ;BR IF SET
```

```
035554  004737  041664                  JSR     PC,SET.IE       ;SET "IE" WITHOUT A "TRE"
035560  012637  177776          8$:     MOV     (SP)+,@#PS      ;RESTORE PROC. STATUS
035564  104413                          RESREG                  ;RESTORE R0 - R5
035566  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

035570  004737  042424          CI1:    JSR     PC,POPQUE       ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
035574  010237  034230                  MOV     R2,TRNSWT       ;PUT REQ. IN TRANSFER WAIT QUEUE
035600  010203                          MOV     R2,R3           ;DPB ADDRESS TO R3
035602  013704  034316                  MOV     RMADR,R4        ;RMCS1 ADDRESS
035606  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
035612  062703  000004                  ADD     #4,R3           ;DESIRED WORD COUNT
035616  062704  000002                  ADD     #2,R4           ;RMWC ADDRESS
035622  012324                          MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
035624  012324                          MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
035626  012346                          MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
035630  004037  041360                  JSR     R0,WRT.RM       ;CALL THE LOAD(WRITE) ROUTINE
035634  000006                          RMDA                    ;INDEX OF REGISTER TO LOAD
035636  036412                          CI7                     ;ERROR RETURN ADDRESS
035640  012346                          MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
035642  004037  041360                  JSR     R0,WRT.RM
035646  000034                          RMDC
035650  036412                          CI7
035652  016246  000002                  MOV     2(R2),-(SP)     ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
035656  004037  041360                  JSR     R0,WRT.RM
035662  000000                          RMCS1
035664  036412                          CI7
035666  010137  034302                  MOV     R1,DTUW         ;SET "DATA TRANSFER UNDERWAY"
035672  000137  036354                  JMP     CI5
035676  013704  034316          CI3:    MOV     RMADR,R4        ;RMCS1 ADDRESS
035702  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
035706  016246  000012                  MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
035712  004037  041360                  JSR     R0,WRT.RM
035716  000034                          RMDC
035720  036412                          CI7
035722  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
035726  010346                  1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
035730  042716  177740                  BIC     #177740,(SP)    ;CHOP OF ALL EXENTED BITS ,IF ANY
035734  116266  000011  000001          MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
035742  004037  041360                  JSR     R0,WRT.RM       ;LOAD DESIRED TRACK & SECTOR
035746  000006                          RMDA
035750  036412                          CI7
035752  012746  000131                  MOV     #131,-(SP)      ;START A SEARCH
```

```
035756  004037  041360                  JSR     R0,WRT.RM
035762  000000                          RMCS1
035764  036412                          CI7
035766  156137  034304  034232          BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
035774  000567                          BR      CI5
035776  013704  034316          CI4:    MOV     RMADR,R4        ;RMCS1 ADDRESS
036002  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
036006  116203  000002                  MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
036012  122703  000131                  CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
036016  001007                          BNE     1$              ;BRANCH IF NO
036020  016246  000010                  MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
036024  004037  041360                  JSR     R0,WRT.RM
036030  000006                          RMDA
036032  036412                          CI7
036034  000403                          BR      2$              ;GO LOAD CYLINDER
036036  122703  000105          1$:     CMPB    #105,R3         ;IS IT A SEEK COMMAND
036042  001007                          BNE     3$              ;BRANCH IF NO
036044  016246  000012          2$:     MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
036050  004037  041360                  JSR     R0,WRT.RM
036054  000034                          RMDC
036056  036412                          CI7
036060  000546                          BR      CI6
036062  122703  000115          3$:     CMPB    #115,R3         ;IS IT AN "OFFSET" COMMAND?
036066  001013                          BNE     4$              ;BR IF NO
036070  004037  041204                  JSR     R0,RD.RM        ;MERGE THE OFFSET VALUE INTO RMOF
036074  000032                          RMOF                    ;BUT DON'T CHANGE THE UPPER
036076  036412                          CI7
036100  116216  000001                  MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
036104  004037  041360                  JSR     R0,WRT.RM       ;REGISTER (RMOF)
036110  000032                          RMOF
036112  036412                          CI7
036114  000530                          BR      CI6             ;GO START THE COMMAND
036116  122703  000107          4$:     CMPB    #107,R3         ;IS IT A "RECALIBRATE" COMMAND?
036122  001525                          BEQ     CI6             ;BRANCH IF YES
036124  122703  000117                  CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
036130  001522                          BEQ     CI6             ;BRANCH IF YES
036132  122703  000103                  CMPB    #103,R3         ;IS IT AN "UNLOAD" COMMAND?
036136  001016                          BNE     5$              ;BRANCH IF NO
036140  112761  000001  034160          MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
036146  105061  034170                  CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
036152  112761  000001  034236          MOVB    #1,ULDFLG(R1)   ;SET "UNLOAD IN PROGRESS" FLAG
036160  010346                          MOV     R3,-(SP)        ;START THE "UNLOAD" COMMAND
036162  004037  041360                  JSR     R0,WRT.RM
036166  000000                          RMCS1
036170  036412                          CI7
036172  000207                          RTS     PC              ;RETURN TO USER
036174  122703  000143          5$:     CMPB    #143,R3         ;IS IT A "SET FORMAT" COMMAND?
036200  001014                          BNE     6$              ;BRANCH IF NO
036202  004037  041204                  JSR     R0,RD.RM        ;READ THE OFFSET REGISTER
036206  000032                          RMOF
036210  036412                          CI7
036212  116266  000001  000001          MOVB    1(R2),1(SP)     ;COMBINE "FMT22","ECI", AND "HCI"
036220  004037  041360                  JSR     R0,WRT.RM       ;LOAD "FMT22", "ECI", AND/OR "HCI".
036224  000032                          RMOF
036226  036412                          CI7
036230  000436                          BR      12$
```

```
036232  122703  000141          6$:     CMPB    #141,R3          ;IS IT A "GET REGISTER" COMMAND?
036236  001023                          BNE     10$              ;BRANCH IF NO
036240  016203  000006          7$:     MOV     6(R2),R3         ;POINTS TO 1ST ADDRESS OF WHERE
                                                                 ;TO PUT THE REGISTER(S)
036244  116237  000010  036262          MOVB    10(R2),9$        ;INIT. THE INDEX FOR THE FIRST REG.
036252  116205  000011                  MOVB    11(R2),R5        ;INDEX OF LAST REG. TO MOVE
036256  004037  041204          8$:     JSR     R0,RD.RM         ;READ RH70/RM03 REGISTER
036262  000000                  9$:     RMCS1                    ;INDEX OF REG. TO READ
036264  036412                          CI7
036266  012623                          MOV     (SP)+,(R3)+      ;GET THE CONTENTS OF RH70//RM03 REG.
036270  023705  036262                  CMP     9$,R5            ;LAST REG. BEEN READ?
036274  001414                          BEQ     12$              ;GET OUT IF YES
036276  062737  000002  036262          ADD     #2,9$            ;INCREASE THE INDEX BY 2
036304  000764                          BR      8$               ;LOOP--MORE TO READ
036306  122703  000145          10$:    CMPB    #145,R3          ;IS IT A "SELECT DRIVE" COMMAND?
036312  001405                          BEQ     12$              ;BRANCH IF YES
036314  010346                  11$:    MOV     R3,-(SP)         ;LOAD THE COMMAND
036316  004037  041360                  JSR     R0,WRT.RM
036322  000000                          RMCS1
036324  036412                          CI7
036326  004737  042424          12$:    JSR     PC,POPQUE        ;REMOVE REQ. FROM QUEUE
036332  052762  000200  000016          BIS     #BIT07,16(R2)    ;SET THE "DONE" BIT
036340  005737  034256                  TST     SAVEFG           ;SAVE THE RH70/RM03 REGISTERS?
036344  100002                          BPL     13$              ;BRANCH IF NO
036346  004737  041546                  JSR     PC,SVRH70        ;YES--GO SAVE THE REGISTERS
036352  000207                  13$:    RTS     PC               ;RETURN TO USER
036354  006301                  CI5:    ASL     R1
036356  012761  001750  034262          MOV     #1000.,TIMER(R1)         ;SET A ONE SECOND TIMER
036364  006201                          ASR     R1
036366  112761  000001  034160          MOVB    #1,DRVACT(R1)    ;SET THE DRIVE ACTIVE
036374  000207                          RTS     PC               ;RETURN TO THE USER
036376  010346                  CI6:    MOV     R3,-(SP)         ;LOAD THE COMMAND
036400  004037  041360                  JSR     R0,WRT.RM
036404  000000                          RMCS1
036406  036412                          CI7
036410  000761                          BR      CI5
036412  032764  010000  000010  CI7:    BIT     #BIT12,RMCS2(R4)  ;DRIVE NON-EXISTENT ?
036420  001034                          BNE     CI8              ;BR IF YES
036422  005702                  1$:     TST     R2               ;ANYTHING IN QUEUE ?
036424  001405                          BEQ     CI7B             ;BR IF NOT
036426  012762  104000  000016          MOV     #BIT15!BIT11,16(R2)      ;SET "PARITY" ERROR INDICATOR
036434  004737  041546                  JSR     PC,SVRH70        ;GO SAVE THE RH70/RM03 REGISTERS
036440  012746  000111          CI7B:   MOV     #111,-(SP)       ;DO A "DRIVE CLEAR"
036444  004037  041360                  JSR     R0,WRT.RM
036450  000000                          RMCS1
036452  036512                          CI8
036454  004737  042306                  JSR     PC,EMPTYQ        ;EMPTY THE QUEUE
036460  105061  034236                  CLRB    ULDFLG(R1)       ;CLEAR THE UNLOAD IN QUEUE FLAG
036464  105061  034160                  CLRB    DRVACT(R1)       ;DRIVE IS IDLE
036470  020137  034302                  CMP     R1,DTUW          ;IF THIS DRIVE HAD AN I/O REQUEST
036474  001005                          BNE     1$               ;IN PROGRESS CLEAR ALL OF THE FLAGS
036476  005037  034230                  CLR     TRNSWT
036502  012737  177777  034302          MOV     #-1,DTUW
036510  000207                  1$:     RTS     PC
036512  104412                  CI8:    SAVREG                   ;SAVE R0 - R5
036514  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;IS 'NED' SET ?
```

```
036522  001002                          BNE     1$              ;BR IF YES
036524  005001                          CLR     R1
036526  005003                          CLR     R3
036530  105761  034160          1$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
036534  001443                          BEQ     5$              ;BRANCH IF NO
036536  013702  034230                  MOV     TRNSWT,R2       ;GET THE "TRANSFER WAIT" QUEUE
036542  020137  034302                  CMP     R1,DTUW         ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
036546  001402                          BEQ     2$              ;BRANCH IF YES
036550  004737  042402                  JSR     PC,GETREQ       ;GET THE DPB POINTER
036554  005702                  2$:     TST     R2              ;QUEUE ENTRY FOR DRIVE ?
036556  001415                          BEQ     4$              ;BR IF NOT
036560  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
036566  001404                          BEQ     3$              ;BR IF NOT
036570  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
036576  000405                          BR      4$              ;CONTINUE
036600  012762  102000  000016  3$:     MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
036606  004737  041546                  JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
036612  012763  177777  034262  4$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
036620  105061  034160                  CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
036624  020137  034302                  CMP     R1,DTUW         ;IS THIS DRIVE SETUP FOR A TRANSFER
036630  001005                          BNE     5$              ;BR IF NOT
036632  012737  177777  034302          MOV     #-1,DTUW        ;RESET THE INDICATOR
036640  005037  034230                  CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
036644  105061  034236          5$:     CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
036650  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
036656  001021                          BNE     6$              ;BR IF YES
036660  005201                          INC     R1              ;MOVE TO THE NEXT DRIVE
036662  062703  000002                  ADD     #2,R3
036666  042701  177770                  BIC     #^C7,R1
036672  001316                          BNE     1$              ;BRANCH IF MORE DRIVES
036674  012737  177777  034302          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
036702  005037  034230                  CLR     TRNSWT          ;CLEAR THE 'TRANSFER WAIT' QUEUE
036706  004737  042230                  JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
036712  012764  000040  000010          MOV     #BIT05,RMCS2(R4) ;DO A MASSBUS INIT.
036720  000406                          BR      7$              ;CONTINUE
036722  004737  042306          6$:     JSR     PC,EMPTYQ       ;CLEAR THE DRIVE'S QUEUE
036726  105061  034170                  CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
036732  105061  034200                  CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
036736  004737  041664          7$:     JSR     PC,SET.IE       ;SET  IE" WITHOUT "TRE"
036742  104413                          RESREG                  ;RESTORE R0 - R5
036744  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

036746  013704  034316          LA:     MOV     RMADR,R4        ;GET RMCS1'S ADDRESS
036752  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
036756  004037  041204                  JSR     R0,RD.RM        ;READ DRIVE STATUS
036762  000012                          RMDS
```

```
036764  037114                          4$                          ;ERROR RETURN ADDRESS
036766  042716  157577                  BIC     #↑C020200,(SP)      ;ON CYLINDER ?
036772  022726  000200                  CMP     #200,(SP)+          ;PIP=0,DRY=1?
036776  001044                          BNE     3$                  ;NO
037000  105261  034246                  INCB    LACNT(R1)           ;INCREMENT THE LOOK AHEAD COUNT
037004  126137  034246  034324          CMPB    LACNT(R1),MXLACT    ;EXCEED MAX?
037012  003033                          BGT     2$                  ;BRANCH IF YES
037014  116203  000010                  MOVB    10(R2),R3           ;GET DESIRED SECTOR ADDRESS AND
037020  000303                          SWAB    R3                  ;MULT. BY 64--ALIGN WITH
037022  006203                          ASR     R3                  ;LOOK AHEAD REGISTER
037024  006203                          ASR     R3
037026  012737  000340  177776          MOV     #340,@#PS           ;PRIORITY LEVEL "7"
037034  004037  041204          6$:     JSR     R0,RD.RM            ;READ LOOK AHEAD REGISTER
037040  000020                          RMLA
037042  037114                          4$
037044  021664  000020                  CMP     (SP),RMLA(R4)       ;CORRECT LA NUMBER ?
037050  001402                          BEQ     7$                  ;YES
037052  005726                          TST     (SP)+               ;NO,CLEAR STACK
037054  000415                          BR      3$
037056  162603                  7$:     SUB     (SP)+,R3            ;CALCULATE THE DELTA
037060  002002                          BGE     1$
037062  062703  004000                  ADD     #<32.*64.>,R3       ,MAKE THE DELTA POSITIVE
037066  023703  034326          1$:     CMP     MXDLTA,R3           ;CHECK THE DELTA TO SEE
037072  002406                          BLT     3$                  ;IF IT IS WITHIN THE
037074  023703  034330                  CMP     MNDLTA,R3           ;WINDOW---IF YES, ZERO
037100  002003                          BGE     3$                  ;THE LOOK AHEAD COUNT
037102  105061  034246          2$:     CLRB    LACNT(R1)           ;AND TAKE THE I/O EXIT
037106  005720                          TST     (R0)+
037110  005720                  3$:     TST     (R0)+               ;ADJUST THE RETURN ADDRESS
037112  000402                          BR      5$                  ;EXIT
037114  004737  036412          4$:     JSR     PC,CI7              ;PROCESS THE ERROR
037120  000200                  5$:     RTS     R0                  ;RETURN

                                ;INTERRUPT SERVICE ROUTINE

037122  112737  000001  034234  ISR:    MOVB    #1,ACTDRV           ;SET "ACTIVE DRIVER" FLAG
037130  104412                          SAVREG                      ;SAVE R0 - R5
037132  013704  034316                  MOV     RMADR,R4            ;ADDRESS OF RHSCS1
037136  013701  034302                  MOV     DTUW,R1             ;GET "DATA TRANSFER UNDERWAY" INDICATOR
037142  002403                          BLT     1$                  ;BRANCH IF NO DATA TRANSFER UNDERWAY
037144  004737  037166                  JSR     PC,TD               ;CALL TRANSFER DONE
037150  000402                          BR      2$                  ;EXIT
037152  004737  037442          1$:     JSR     PC,SC               ;CALL SPECIAL CONDITIONS
037156  104413                  2$:     RESREG                      ;RESTORE R0 - R5
037160  105037  034234                  CLRB    ACTDRV              ;CLEAR "ACTIVE DRIVER" FLAG
037164  000002                          RTI                         ;RETURN

                                ;TRANSFER DONE ROUTINE

037166  105061  034160          TD:     CLRB    DRVACT(R1)          ;SET DRIVE ACTIVE INDICATOR TO IDLE
037172  012737  177777  034302          MOV     #-1,DTUW            ;NO DATA TRANSFERS UNDERWAY
037200  006301                          ASL     R1
037202  012761  177777  034262          MOV     #-1,TIMER(R1)       ;CANCEL TIMEOUT
037210  006201                          ASR     R1
037212  013702  034230                  MOV     TRNSWT,R2           ;GET "DPB" ADDRESS FROM THE
037216  005037  034230                  CLR     TRNSWT              ;TRANSFER WAIT QUEUE--CLEAR QUEUE
```

```
037222  052762  000200  000016          BIS     #BIT07,16(R2)    ;SET DONE
037230  010164  000010                  MOV     R1,RMCS2(R4)     ;SELECT THE DRIVE
037234  004037  041204                  JSR     R0,RD.RM         ;TRANSFER ERROR(TRE=1)?
037240  000000                          RMCS1
037242  036412                          CI7
037244  006126                          ROL     (SP)+
037246  100421                          BMI     3$               ;BR IF YES
037250  005737  034256                  TST     SAVEFG           ;SAVE THE RH70/RM03 REGISTERS?
037254  100002                          BPL     1$               ;BRANCH IF NO
037256  004737  041546                  JSR     PC,SVRH70        ;YES--SAVE THE REGISTERS
037262  004737  037342          1$:     JSR     PC,WC.HK                 ;SEE IF WRITE CHECK TO BE PUT IN QUEUE
037266  004737  042402                  JSR     PC,GETREQ        ;GET DPB POINTER
037272  005702                          TST     R2               ;ENTRY FOR DRIVE ?
037274  001403                          BEQ     2$               ;BR IF NOT
037276  004737  035332                  JSR     PC,OPT           ;CALL OPTIMIZER
037302  000457                          BR      SC               ;CHECK OTHER DRIVES
                                ;THE RELEASE DRIVE COMMAND IS FORECD TO ENTER FOR DUAL PORT OPERATION
037304  012714  000113          2$:     MOV     #113,(R4)        ;RELEASE THE DRIVE
037310  000454                          BR      SC               ;CHECK FOR OTHER DRIVES
037312  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
037320  004737  042306                  JSR     PC,EMPTYQ        ;EMPTY THE "DRIVE'S WAIT" QUEUE
037324  004737  041546                  JSR     PC,SVRH70        ;SAVE THE RH70/RM03 REGISTERS
037330  012714  040111                  MOV     #40111,(R4)      ;ISSUE A "DRIVE CLEAR"
037334  012714  000113                  MOV     #113,(R4)        ;ISSUE A RELEASE TO THE DRIVE
037340  000440                          BR      SC               ;CHECK FOR OTHER DRIVES


037342  122762  000002  000024  WC.HK:  CMPB    #2,$CODE(R2)     ;LAST OPERATION WRITE DATA ?
037350  001404                          BEQ     1$               ;BR IF IT WAS
037352  122762  000003  000024          CMPB    #3,$CODE(R2)     ;LAST OPERATION WRITE HEADER & DATA ?
037360  001027                          BNE     2$               ;BR IF NOT
037362  004037  042326          1$:     JSR     R0,DRVQUE        ;PUT THE OPERATION IN THE QUEUE
037366  000424                          BR      2$               ;QUEUE IS FULL
037370  005062  000016                  CLR     16(R2)           ;CLEAR 'DONE' BIT IN DPB
037374  116262  000234  000027          MOVB    $RMCS1(R2),$PREV(R2)    ;SAVE WRITE OPERATION CODE
037402  016262  000012  000034          MOV     $CYL(R2),$PREVA+2(R2)   ;SAVE CYLINDER
037410  016262  000010  000032          MOV     $SEC(R2),$PREVA(R2)  ;SAVE SECTOR AND TRACK ADDRESSES
037416  142762  000002  000024          BICB    #2,$CODE(R2)     ;CHANGE WRITE TO CHECK
037424  142762  000020  000002          BICB    #20,$COMND(R2)   ;CHANGE DRIVER CODE TO WRITE CHECK
037432  152762  000010  000002          BISB    #10,$COMND(R2)   ;FINISH CHANGING CODE TO WRITE CHECK
037440  000207                  2$:     RTS     PC               ;EXIT


                                ;SPECIAL CONDITION ROUTINE

037442  116403  000016          SC:     MOVB    RMAS(R4),R3      ;READ "RMAS"
037446  001012                          BNE     2$               ;BRANCH IF ANY 'ATA' BITS SET
037450  004037  041204                  JSR     R0,RD.RM         ;READ CONTROL AND STATUS REGISTER
037454  000000                          RMCS1
037456  036512                          CI8
037460  106126                          ROLB    (SP)+            ;IS "IE"=1?
037462  100403                          BMI     1$               ;YES, NO DRIVES TO CHECK
037464  104001                          ERROR   1                ;REPORT AN ILLEGAL INTERRUPT
037466  004737  041664                  JSR     PC,SET.IE        ;SET INTERRUPT ENABLE
037472  000207                  1$:     RTS     PC               ;RETURN
037474  005046                  2$:     CLR     -(SP)            ;PROCESS ALL DRIVES THAT HAVE
037476  110316                          MOVB    R3,(SP)          ;AN "ATA"=1
```

```
037500  012703  000001                  MOV     #1,R3
037504  005001                          CLR     R1
037506  030316                  SC3:    BIT     R3,(SP)         ;ATA=1?
037510  001005                          BNE     SC5             ;YES--BRANCH
037512  005201                  SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
037514  106303                          ASLB    R3
037516  001373                          BNE     SC3             ;BRANCH IF MORE TO CHECK?
037520  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
037522  000207                          RTS     PC              ;RETURN TO USER
037524  105761  034210          SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
037530  001402                          BEQ     1$              ;BR IF NOT
037532  000137  040440                  JMP     SC13            ;PROCESS THE DRIVE
037536  105761  034220          1$:     TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
037542  001402                          BEQ     2$              ;BR IF NOT
037544  000137  040440                  JMP     SC13            ;START THE OUTSTANDING COMMAND
037550  105761  034170          2$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
037554  003025                          BGT     5$              ;BRANCH IF ONLINE
037556  105761  034236                  TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
037562  003422                          BLE     5$              ;BRANCH IF NOT
037564  004737  042402                  JSR     PC,GETREQ       ;GET DPB POINTER
037570  004737  041546                  JSR     PC,SVRH70       ;SAVE THE RH70/RM03 REGISTERS
037574  004737  040370                  JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                ;ALSO DO A DRIVE INIT (DRVINT)
037600  105761  034170                  TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
037604  003416                          BLE     6$              ;NO---BRANCH
037606  032737  040000  034150          BIT     #BIT14,RMERRS   ;WAS THERE AN ERROR?
037614  001002                          BNE     3$              ;BR IF ERROR
037616  000137  040260                  JMP     SC11            ;NO ERROR
037622  013705  034152          3$:     MOV     RMERRS+2,R5     ;YES -- PICKUP RMER1 AND
037626  000476                          BR      SC6A            ;GO PROCESS THE ERROR
037630  105761  034160          5$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
037634  001027                          BNE     SC6             ;BR IF EITHER
037636  004737  040370                  JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                ;ALSO DO A DRVINT
037642  105761  034210          6$:     TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
037646  001321                          BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
037650  105761  034170                  TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
037654  100412                          BMI     7$              ;BR IF UNSAFE
037656  032737  020000  034154          BIT     #BIT13,RMERRS+4 ;ADDRESS PLUG CHANGED ?
037664  001011                          BNE     8$              ;BR IF YES
037666  012746  000113                  MOV     #113,-(SP)      ;RELEASE COMMAND
037672  004037  041360                  JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RMCS1
037676  000000                          RMCS1                   ;REGISTER INDEX
037700  040230                          SC8                     ;PARITY EXIT ADDRESS
037702  011605                  7$:     MOV     (SP),R5         ;PICKUP (RMAS) BEFORE THE ERROR CALL
037704  104002                          ERROR   2               ;REPORT THE UNEXPECTED ATTENTION
037706  000701                          BR      SC4             ;GO CHECK FOR MORE ATA'S
037710                          8$:
037710  104005                          ERROR   5               ;REPORT THE ADDRESS PLUG CHANGE
037712  000677                          BR      SC4             ;CHECK FOR MORE DRIVES
037714  006301                  SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
037716  012761  177777  034262          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
037724  006201                          ASR     R1              ;RESTORE THE DRIVE ADDRESS
037726  004737  042402                  JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
037732  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
037736  004037  041204                  JSR     R0,RD.RM        ;READ THE RM03'S STATUS REG.
```

```
037742  000012                          RMDS
037744  040230                          SC8
037746  011605                          MOV     (SP),R5         ;AND PUT IT IN R5
037750  006126                          ROL     (SP)+           ;WAS THERE AN ERROR?
037752  100407                          BMI     1$              ;BR IF ERROR
037754  105761  034160                  TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
037760  003137                          BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
037762  052762  100210  000016          BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
037770  000470                          BR      SC7
037772  004037  041204          1$:     JSR     R0,RD.RM        ;READ ERROR REGISTER #1
037776  000014                          RMER1
040000  040230                          SC8
040002  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
040004  004737  041546                  JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
040010  012746  000111                  MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
040014  004037  041360                  JSR     R0,WRT.RM
040020  000000                          RMCS1
040022  040230                          SC8
040024  006105                  SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
040026  100406                          BMI     1$              ;BRANCH IF YES
040030  005702                          TST     R2              ;ANYTHING IN QUEUE ?
040032  001447                          BEQ     SC7             ;BR IF NOT
040034  052762  100240  000016          BIS     #BIT15!BIT07!BIT05,16(R2)      ;INFORM USER OF ERROR
040042  000443                          BR      SC7
040044  004037  041204          1$:     JSR     R0,RD.RM        ;READ DRIVE STATUS REG. #1
040050  000012                          RMDS
040052  040230                          SC8
040054  011605                          MOV     (SP),R5         ;SAVE RMDS IN R5
040056  006126                          ROL     (SP)+           ;"ERR"=1?
040060  100011                          BPL     2$              ;BR IF NO--UNSAFE CLEARED
040062  112761  177777  034170          MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
040070  004737  041546                  JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
040074  052762  110000  000016          BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
040102  000423                          BR      SC7
040104  032705  010000          2$:     BIT     #BIT12,R5       ;"MOL" = 1 ?
040110  001015                          BNE     3$              ;BR IF YES
040112  112761  177777  034160          MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
040120  112761  000001  034170          MOVB    #1,DRVSTA(R1)   ;ONLINE
040126  006301                          ASL     R1
040130  012761  072460  034262          MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
040136  006201                          ASR     R1
040140  000137  037512                  JMP     SC4
040144  052762  100220  000016  3$:     BIS     #BIT15!BIT07!BIT04,16(R2)      ;INFORM USER OF ERROR
040152  105061  034160          SC7:    CLRB    DRVACT(R1)      ;DRIVE IS IDLE
040156  004737  042306                  JSR     PC,EMPTYQ       ;DUMP THE QUEUE
040162  105761  034236                  TSTB    ULDFLG(R1)      ;UNLOAD IN RMOGRESS OR QUEUE?
040166  003002                          BGT     1$              ;BR IF NOT
040170  105061  034236                  CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
040174  116164  034304  000016  1$:     MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR ATTENTION BIT
040202  105761  034170                  TSTB    DRVSTA(R1)      ;IS THE DRIVE UNSAFE ?
040206  100406                          BMI     2$              ;BR IF IT IS
040210  012746  000113                  MOV     #113,-(SP)      ;RELEASE COMMAND
040214  004037  041360                  JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RPCS1
040220  000000                          RMCS1                   ;REGISTER INDEX
040222  040230                          SC8                     ;PARITY EXIT ADDRESS
040224  000137  037512          2$:     JMP     SC4             ;CHECK FOR MORE DRIVES
```

```
040230  105761  034160          SC8:    TSTB    DRVACT(R1)          ;IS DRIVE IDLE?
040234  001405                          BEQ     1$                  ;YES--BRANCH
040236  004737  042402                  JSR     PC,GETREQ           ;GET DPB POINTER
040242  004737  036412                  JSR     PC,CI7              ;PROCESS THE PARITY ERROR
040246  000402                          BR      2$                  ;CONTINUE
040250  004737  036440          1$:     JSR     PC,CI7B             ;PROCESS THE UNCORRECTABLE PARITY ERROR
040254  000137  037512          2$:     JMP     SC4                 ;CHECK MORE DRIVES
040260  105761  034236          SC11:   TSTB    ULDFLG(R1)          ;"UNLOAD IN PROGRESS"?
040264  003402                          BLE     1$                  ;BRANCH IF NO
040266  105061  034236                  CLRB    ULDFLG(R1)          ;CLEAR UNLOAD FLAG
040272  105061  034160          1$:     CLRB    DRVACT(R1)          ;SET DRIVE IDLE
040276  136137  034304  034232          BITB    ATABIT(R1),SRCHWT   ;DOING A SEARCH OPERATION FOR
                                                                    ;AN I/O COMMAND?
040304  001012                          BNE     2$                  ;BRANCH IF YES
040306  004737  042424                  JSR     PC,POPQUE           ;REMOVE REQUEST FROM QUEUE
040312  052762  000200  000016          BIS     #BIT07,16(R2)       ;SET "DONE" BIT
040320  005737  034256                  TST     SAVEFG              ;SAVE THE REGISTERS?
040324  100002                          BPL     2$                  ;BRANCH IF NO
040326  004737  041546                  JSR     PC,SVRH70           ;YES--SAVE ALL OF THE RH70/RM03 REG'S
040332  116164  034304  000016  2$:     MOVB    ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
040340  146137  034304  034232          BICB    ATABIT(R1),SRCHWT       ;CLEAR IMPLIED SEEK SET
040346  006301                          ASL     R1                  ;WORD INDEX
040350  012761  177777  034262          MOV     #-1,TIMER(R1)       ;STOP CLOCK
040356  006201                          ASR     R1                  ;RESTORE R1
040360  004737  035332                  JSR     PC,OPT              ;START A REQUEST
040364  000137  037512                  JMP     SC4                 ;CHECK FOR MORE DRIVES
040370  010164  000010          SC12:   MOV     R1,RMCS2(R4)        ;SELECT DRIVE
040374  016437  000012  034150          MOV     RMDS(R4),RMERRS     ;SAVE THE FOUR REGISTERS THAT
040402  016437  000014  034152          MOV     RMER1(R4),RMERRS+2      ;WILL TELL US SOMETHING
040410  016437  000042  034154          MOV     RMER2(R4),RMERRS+4
040416  016437  000040  034156          MOV     RMMR2(R4),RMERRS+6
040424  004037  034546                  JSR     R0,DRVINT           ;INIT. THE STATE OF THE DRIVE
040430  000401                          BR      1$                  ;TAKE ERROR EXIT
040432  000207                          RTS     PC                  ;RETURN
040434  005726                  1$:     TST     (SP)+               ;POP PC OFF OF THE STACK
040436  000674                          BR      SC8                 ;PROCESS THE PARITY ERROR
040440  006301                  SC13:   ASL     R1                  ;SETUP TO ADDRESS WORDS
040442  012761  177777  034262          MOV     #-1,TIMER(R1)       ;STOP THE TIMER
040450  006201                          ASR     R1                  ;
040452  010164  000010                  MOV     R1,RMCS2(R4)        ;SELECT THE DRIVE
040456  116164  034304  000016          MOVB    ATABIT(R1),RMAS(R4) ;CLEAR THE ATTENTION BIT
040464  105761  034210          1$:     TSTB    DPINT(R1)           ;INITIALIZING THE DRIVE ?
040470  001424                          BEQ     2$                  ;BR IF NOT
040472  105061  034210                  CLRB    DPINT(R1)           ;CLEAR THE INIT INDICATOR
040476  004037  034546                  JSR     R0,DRVINT           ;GO INIT THE DRIVE
040502  000240                          NOP                         ;DUMMY PARITY ERROR RETURN
040504  105761  034170                  TSTB    DRVSTA(R1)          ;DRIVE ONLINE ?
040510  003014                          BGT     2$                  ;BR IF YES -- START ORDER
040512  005702                          TST     R2                  ;QUEUE ENTRY FOR THE DRIVE
040514  001426                          BEQ     3$                  ;BR IF NOT
040516  004737  042402                  JSR     PC,GETREQ           ;GET DPB ADDRESS
040522  052762  140000  000016          BIS     #BIT15!BIT14,16(R2) ;INFORM USER THAT DRIVE OFFLINE
040530  004737  041546                  JSR     PC,SVRH70           ;SAVE THE REGISTERS
040534  004737  042306                  JSR     PC,EMPTYQ           ;EMPTY THE REQUEST QUEUE
040540  000414                          BR      3$
040542  032764  004000  000000  2$:     BIT     #BIT11,RMCS1(R4)        ;DVA SET ?
```

```
040550  001006                          BNE     4$              ;SET THEN CALL OPT
040552  006301                          ASL     R1
040554  012761  023420  034262          MOV     #10000.,TIMER(R1)
040562  006201                          ASR     R1
040564  000402                          BR      3$
040566  004737  035332          4$:     JSR     PC,OPT          ;START THE PENDING REQUEST
040572  000137  037512          3$:     JMP     SC4             ;PROCESS OTHER DRIVES

                                ;/RM03 TIMER ROUTINE
                                ;CALL
                                ;       MOV     #TIME,-(SP)     ;ELASPED TIME IN MILLISECONDS ON THE STACK
                                ;       JSR     PC,RPTMR        ;CALL RM03 TIME ROUTINE

040576  005737  034234          RPTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
040602  001027                          BNE     4$              ;IF NON ZERO EXIT
040604  112737  000001  034235          MOVB    #1,ACTSTR       ;SET "ACTSTR"
040612  104412                          SAVREG                  ;SAVE R0 - R5
040614  005001                          CLR     R1              ;START WITH DRIVE 0
040616  005003                          CLR     R3
040620  005763  034262          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
040624  002406                          BLT     2$              ;BRANCH IF NO
040626  166663  000002  034262          SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
040634  003002                          BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
040636  004737  040666                  JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
040642  005201                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
040644  005723                          TST     (R3)+
040646  022701  000010                  CMP     #8.,R1          ;OUT OF DRIVES?
040652  003362                          BGT     1$              ;BRANCH IF NO
040654  104413                  3$:     RESREG                  ;RESTORE R0 - R5
040656  105037  034235                  CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
040662  012616                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
040664  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

040666  010146                  STO:    MOV     R1,-(SP)        ;SAVE R1
040670  010246                          MOV     R2,-(SP)        ;SAVE R2
040672  010346                          MOV     R3,-(SP)        ;SAVE R3
040674  010446                          MOV     R4,-(SP)        ;SAVE R4
040676  013704  034316                  MOV     RMADR,R4        ;GET ADDRESS OF "RMCS1"
040702  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
040706  004037  041204                  JSR     R0,RD.RM        ;READ "DRIVE STATUS REG"
040712  000012                          RMDS
040714  041066                          STOS
040716  105726                          TSTB    (SP)+           ;IS "DRY"=1?
040720  100434                          BMI     STO2            ;BR IF YES
040722  105761  034210          STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
040726  001031                          BNE     STO2            ;BR IF YES
```

```
040730  105761  034220                  TSTB    DPRQS(R1)          ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
040734  001026                          BNE     ST02               ;BR IF YES
040736  013702  034230                  MOV     TRNSWT,R2          ;PICKUP TRANSFER WAIT QUEUE
040742  020137  034302                  CMP     R1,DTUW            ;TRANSFER UNDERWAY ON THIS DRIVE?
040746  001404                          BEQ     1$                 ;BRANCH IF YES
040750  000137  041172                  JMP     ST09               ;IF NOT DON'T BOTHER DRIVES
040754  004737  042402                  JSR     PC,GETREQ          ;GET DPB ADDRESS
040760  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
040766  004737  041546                  JSR     PC,SVRH70          ;SAVE RH70/RM03 REGISTERS
040772  012764  000040  000010          MOV     #BIT05,RMCS2(R4)   ;"INIT" THE MASS BUS
041000  105061  034160                  CLRB    DRVACT(R1)         ;DRIVE IS IDLE
041004  105061  034236                  CLRB    ULDFLG(R1)         ;CLEAR THE UNLOAD FLAG
041010  000470                          BR      ST09               ;DON'T BOTHER OTHER DRIVES
041012  116405  000016          ST02:   MOVB    RMAS(R4),R5        ;READ ATTENTION REG
041016  136105  034304                  BITB    ATABIT(R1),R5      ;IS ATTENTION FOR THIS DRIVE UP ?
041022  001007                          BNE     ST03               ;YES--BRANCH
041024  105761  034210                  TSTB    DPINT(R1)          ;TRYING TO INTIALIZE THE DRIVE ?
041030  001021                          BNE     ST06               ;BR IF YES - DRIVE NOT ONLINE
041032  105761  034220                  TSTB    DPRQS(R1)          ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
041036  001035                          BNE     ST07               ;BR IF YES - NO RESPONSE TO REQUEST
041040  000454                          BR      ST09               ;OTHER WISE EXIT
041042  105761  034210          ST03:   TSTB    DPINT(R1)          ;INITIALIZING THE DRIVE ?
041046  001003                          BNE     1$                 ;BR IF INIT PENDING
041050  105761  034220                  TSTB    DPRQS(R1)          ;PORT REQUEST PENDING ?
041054  001446                          BEQ     ST09               ;BR IF NOT
041056  012763  177777  034262  1$:     MOV     #-1,TIMER(R3)      ;STOP THE TIMER
041064  000442                          BR      ST09               ;EXIT
041066  004737  036512          ST05:   JSR     PC,CI8             ;GO HANDLE THE PARITY ERROR
041072  000437                          BR      ST09
041074  105061  034210          ST06:   CLRB    DPINT(R1)          ;CLEAR THE INITIALIZE INDICATOR
041100  105061  034170                  CLRB    DRVSTA(R1)         ;SET UNIT OFFLINE
041104  012763  177777  034262          MOV     #-1,TIMER(R3)      ;STOP THE TIMER
041112  004737  042402                  JSR     PC,GETREQ          ;GET THE DPB ADDRESS
041116  005702                          TST     R2                 ;REQUEST IN QUEUE ?
041120  001424                          BEQ     ST09               ;BR IF NOT
041122  052762  140000  000016          BIS     #BIT15!BIT14,16(R2) ;INFORM THE USER DRIVE NOT AVAILABLE
041130  000414                          BR      ST08               ;FINISH
041132  012763  177777  034262  ST07:   MOV     #-1,TIMER(R3)      ;STOP THE TIMER
041140  105061  034220                  CLRB    DPRQS(R1)          ;CLEAR PORT REQUEST INDICATOR
041144  004737  042402                  JSR     PC,GETREQ          ;GET DPB ADDRESS
041150  005702                          TST     R2                 ;QUEUE ENTRY FOR DRIVE ?
041152  001407                          BEQ     ST09               ;BR IF NONE
041154  012762  100004  000016          MOV     #BIT15!BIT2,16(R2) ;INFORM USER OF PORT REQUEST ERROR
041162  004737  042306          ST08:   JSR     PC,EMPTYQ          ;CLEAR THE QUEUE FOR THE DRIVE
041166  004737  041546                  JSR     PC,SVRH70          ;SAVE THE REGISTERS
041172  012604                  ST09:   MOV     (SP)+,R4           ;RESTORE R4
041174  012603                          MOV     (SP)+,R3           ;RESTORE R3
041176  012602                          MOV     (SP)+,R2           ;RESTORE R2
041200  012601                          MOV     (SP)+,R1           ;RESTORE R1
041202  000207                          RTS     PC                 ;RETURN

                                ;ROUTINE TO READ A RH70/RM03 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
                                        ;
041204  013737  034314  041346  RD.RM:  MOV     MCPEMX,RD.RM2           ;MAX. RETRYS ALLOWED
041212  011646                          MOV     (SP),-(SP)              ;SAVE R0 FOR RETURN
041214  013737  034316  041230          MOV     RMADR,RD.ADR            ;FORM THE DESIRED ADDRESS
041222  062037  041230                  ADD     (R0)+,RD.ADR            ;USING THE BASE AND THE INDEX
041226  013727                  RD.RM1: MOV     @(PC)+,(PC)+            ;READ THE DESIRED REGISTER OF THE RM03
041230  000000                  RD.ADR: .WORD   0                       ;ADDRESS IS FORMED HERE
041232  000000                  RD.WRD: .WORD   0                       ;REG. CONTENTS PUT HERE
041234  013766  041232  000002          MOV     RD.WRD,2(SP)            ;RETURN IT TO THE USER
041242  013746  034316                  MOV     RMADR,-(SP)             ;PUT THE ADDRESS ON THE STACK
041246  062716  000010                  ADD     #RMCS2,(SP)             ;FORM THE ADDRESS OF RMCS2
041252  032736  010000                  BIT     #BIT12,@(SP)+           ;CHECK THE 'NEI' BIT
041256  001035                          BNE     RD.RM3                  ;BR IF DRIVE NON-EXISTENT
041260  017746  173032                  MOV     @RMADR,-(SP)            ;READ RMCS1
041264  032716  020000                  BIT     #BIT13,(SP)             ;DID MCPE SET?
041270  001002                          BNE     1$                      ;BRANCH IF YES
041272  022620                          CMP     (SP)+,(R0)+             ;ADJUST FOR RETURN
041274  000430                          BR      RD.RM4                  ;EXIT
041276                          1$:
041276  104003                          ERROR   3                       ;REPORT "MCPE" ERROR
041300  005737  034302                  TST     DTUW                    ;DATA TRANSFER UNDERWAY?
041304  100405                          BMI     2$                      ;NO--BRANCH
041306  032716  040000                  BIT     #BIT14,(SP)             ;NO--"TRE"=1?
041312  001402                          BEQ     2$                      ;NO--BRANCH
041314  005726                          TST     (SP)+                   ;YES--CLEAN OFF THE STACK AND
041316  000415                          BR      RD.RM3                  ;TAKE THE FATAL ERROR EXIT
041320  052716  040000          2$:     BIS     #BIT14,(SP)             ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
041324  000316                          SWAB    (SP)                    ;POSITION BEFORE WRITING
041326  013737  034316  041342          MOV     RMADR,3$                ;FORM ADDRESS OF HIGH BYTE
041334  005237  041342                  INC     3$
041340  112637                          MOVB    (SP)+,@(PC)+            ;WRITE THE HIGH BYTE OF RMCS1
041342  000000                  3$:     .WORD   0                       ;ADDRESS STORAGE
041344  005327                          DEC     (PC)+                   ;EXCEEDED MAX. RETRYS
041346  000003                  RD.RM2: .WORD   3
041350  002326                          BGE     RD.RM1                  ;BRANCH IF NO
041352  011000                  RD.RM3: MOV     (R0),R0                 ;FATAL ERROR EXIT
041354  012616                          MOV     (SP)+,(SP)
041356  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

041360  013737  034314  041532  WRT.RM: MOV     MCPEMX,WRT.R2           ;MAX RETRYS ALLOWED
041366  016637  000002  041446          MOV     2(SP),WRT.WD            ;SAVE THE WORD TO WRITE
041374  012616                          MOV     (SP)+,(SP)              ;ADJUST THE STACK
041376  012037  041450                  MOV     (R0)+,WRT.AD            ;GET INDEX OF REGISTER TO BE WRITTEN
041402  001015                          BNE     1$                      ;BRANCH IF NOT RMCS1
```

```
041404  122737  000150  041446          CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
041412  002411                          BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
041414  004037  041204                  JSR     R0,RD.RM        ;NO---COMBINE A16&A17, & PSEL WITH
041420  000000                          RMCS1                   ;THE COMMAND BEFORE SENDING IT TO
041422  041536                          WRT.R3                  ;THE RH70/RM03
041424  000316                          SWAB    (SP)
041426  042716  177770                  BIC     #^C7,(SP)
041432  112637  041447                  MOVB    (SP)+,WRT.WD+1
041436  063737  034316  041450  1$:     ADD     RMADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
041444  012737                  WRT.R1: MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
041446  000000                  WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
041450  000000                  WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
041452  013746  034316                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
041456  062716  000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
041462  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
041466  001023                          BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
041470  004037  041204                  JSR     R0,RD.RM        ;CHECK FOR PARITY ERROR ON WRITE
041474  000014                          RMER1
041476  041536                          WRT.R3
041500  032726  000010                  BIT     #BIT03,(SP)+
041504  001416                          BEQ     WRT.R4          ;BRANCH IF "PAR=0"
041506  016037  177776  041520          MOV     -2(R0),1$       ;PICKUP THE INDEX
041514  004037  041204                  JSR     R0,RD.RM        ;READ THE REG.
041520  000000                  1$:     .WORD   0               ;REG. INDEX
041522  041536                          WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
041524  104004                          ERROR   4               ;REPORT THE PARITY ON WRITE ERROR
041526  005726                          TST     (SP)+           ;CLEAR OFF THE STACK
041530  005327                          DEC     (PC)+           ;DECREMENT THE ERROR COUNT
041532  000003                  WRT.R2: .WORD   3               ;RETRY COUNTER
041534  002343                          BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
041536  011000                  WRT.R3: MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
041540  000401                          BR      WRT.R5          ;EXIT
041542  005720                  WRT.R4: TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
041544  000200                  WRT.R5: RTS     R0

                                ;ROUTINE TO SAVE THE RH70/RP04/5/RM03 REGISTERS AS PER DPB+14
                                ;
                                ;CALL
                                ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
                                ;       JSR     PC,SVRH70       ;SAVE THE DRIVES REG'S

041546  104412                  SVRH70: SAVREG                  ;SAVE R0 - R5
041550  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
041552  001442                          BEQ     6$              ;BR IF NONE
041554  013704  034316                  MOV     RMADR,R4
041560  111264  000010                  MOVB    (R2),RMCS2(R4)  ;SELECT DRIVE
041564  016203  000014                  MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
041570  001433                          BEQ     6$              ;EXIT IF NO ADDRESS
041572  005037  041626                  CLR     3$              ;COUNTER & POINTER
041576  023727  041626  000022  1$:     CMP     3$,#RMDB        ;REACHED THE BUFFER REGISTER ?
041604  001006                          BNE     2$              ;BR IF NOT
041606  032764  000200  000010          BIT     #BIT07,RMCS2(R4)  ;'OR' SET ?
041614  001002                          BNE     2$              ;BR IF SET
041616  005023                          CLR     (R3)+           ;STORE RMDB AS ZEROES
041620  000405                          BR      4$              ;CONTINUE
041622  004037  041204          2$:     JSR     R0,RD.RM        ;READ THE SELECTED REGISTER
```

```
041626  000000                  3$:     .WORD   0               ;REGISTER INDEX
041630  041654                          5$                      ;ERROR RETURN ADDRESS
041632  012623                          MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
041634  023727  041626  000046  4$:     CMP     3$,#RMEC2       ;REACHED THE END ?
041642  001406                          BEQ     6$              ;BR IF YES
041644  062737  000002  041626          ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
041652  000751                          BR      1$              ;CONTINUE READING THE REGISTERS
041654  004737  036412          5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
041660  104413                  6$:     RESREG                  ;RESTORE R0 - R5
041662  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

041664  010446                  SET.IE: MOV     R4,-(SP)        ;SAVE R4
041666  013704  034316                  MOV     RMADR,R4        ;PICKUP ADDRESS OF RMCS1
041672  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
041676  011446                          MOV     (R4),-(SP)      ;READ RMCS1
041700  052716  040000                  BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
041704  000316                          SWAB    (SP)            ;ADJUST FOR DATO
041706  112714  000100                  MOVB    #BIT06,(R4)     ;SET "IE"
041712  032764  010000  000010          BIT     #BIT12,RMCS2(R4) ;IS "NED"=1?
041720  001002                          BNE     1$              ;YES--CLEAR "TRE"
041722  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
041724  000402                          BR      2$
041726  112664  000001          1$:     MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
041732  012604                  2$:     MOV     (SP)+,R4        ;RESTORE R4
041734  000207                          RTS     PC              ;RETURN TO CALLER

                                ;QUEUE COUNT
041736     000                  QCNT:   .BYTE   0               ;DRIVE 0
041737     000                          .BYTE   0               ;DRIVE 1
041740     000                          .BYTE   0               ;DRIVE 2
041741     000                          .BYTE   0               ;DRIVE 3
041742     000                          .BYTE   0               ;DRIVE 4
041743     000                          .BYTE   0               ;DRIVE 5
041744     000                          .BYTE   0               ;DRIVE 6
041745     000                          .BYTE   0               ;DRIVE 7

                                ;QUEUE INPUT POINTERS

041746  042030                  QINPT:  .WORD   QDRV0           ;DRIVE 0
041750  042050                          .WORD   QDRV1           ;DRIVE 1
041752  042070                          .WORD   QDRV2           ;DRIVE 2
041754  042110                          .WORD   QDRV3           ;DRIVE 3
041756  042130                          .WORD   QDRV4           ;DRIVE 4
041760  042150                          .WORD   QDRV5           ;DRIVE 5
041762  042170                          .WORD   QDRV6           ;DRIVE 6
041764  042210                          .WORD   QDRV7           ;DRIVE 7

                                ;QUEUE OUTPUT POINTERS

041766  042030                  QOUTPT: .WORD   QDRV0           ;DRIVE 0
```

```
041770  042050                          .WORD   QDRV1           ;DRIVE 1
041772  042070                          .WORD   QDRV2           ;DRIVE 2
041774  042110                          .WORD   QDRV3           ;DRIVE 3
041776  042130                          .WORD   QDRV4           ;DRIVE 4
042000  042150                          .WORD   QDRV5           ;DRIVE 5
042002  042170                          .WORD   QDRV6           ;DRIVE 6
042004  042210                          .WORD   QDRV7           ;DRIVE 7

042006  042030                  QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
042010  042050                  QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
042012  042070                          .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
042014  042110                          .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
042016  042130                          .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
042020  042150                          .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
042022  042170                          .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
042024  042210                          .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
042026  042230                          .WORD   QTERM           ;STOP DRIVE 7

                                ;DRIVE REQUEST QUEUES

042030  000010                  QDRV0:  .BLKW   10
042050  000010                  QDRV1:  .BLKW   10
042070  000010                  QDRV2:  .BLKW   10
042110  000010                  QDRV3:  .BLKW   10
042130  000010                  QDRV4:  .BLKW   10
042150  000010                  QDRV5:  .BLKW   10
042170  000010                  QDRV6:  .BLKW   10
042210  000010                  QDRV7:  .BLKW   10
        042230                  QTERM=.

                                ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
                                ;
                                ;CALL
                                ;       JSR     PC,CLRQUE

042230  104412                  CLRQUE: SAVREG                  ;SAVE R0 - R5
042232  012702  041736                  MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
042236  005022                          CLR     (R2)+           ;DRIVES 0 & 1
042240  005022                          CLR     (R2)+           ;DRIVES 2 & 3
042242  005022                          CLR     (R2)+           ;DRIVES 4 & 5
042244  005022                          CLR     (R2)+           ;DRIVES 6 & 7
042246  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING
042252  012701  042006                  MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
042256  012122                  1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
042260  005303                          DEC     R3
042262  001375                          BNE     1$
042264  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
042270  012701  042006                  MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
042274  012122                  2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
042276  005303                          DEC     R3
042300  001375                          BNE     2$
042302  104413                          RESREG                  ;RESTORE R0 - R5
042304  000207                          RTS     PC

                                ;EMPTY THE QUEUE SPECIFIED BY R1
                                ;
```

```
                                        ;CALL
                                        ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
                                        ;       JSR     PC,EMPTYQ
042306  105061  041736          EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
042312  006301                          ASL     R1
042314  016161  041746  041766          MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
042322  006201                          ASR     R1
042324  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

042326  122761  000010  041736  DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
042334  001421                          BEQ     2$              ;BR IF YES-TAKE RETURN1
042336  105261  041736                  INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
042342  006301                          ASL     R1
042344  010271  041746                  MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
042350  062761  000002  041746          ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
042356  026161  041746  042010          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
042364  001003                          BNE     1$              ;BRANCH IF NO
042366  016161  042006  041746          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
042374  006201                  1$:     ASR     R1
042376  005720                          TST     (R0)+           ;TAKE RETURN 2
042400  000200                  2$:     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

042402  005002                  GETREQ: CLR     R2
042404  105761  041736                  TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
042410  001404                          BEQ     2$              ;NO---BRANCH
042412  006301                  1$:     ASL     R1
042414  017102  041766                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
042420  006201                          ASR     R1
042422  000207                  2$:     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

042424  105361  041736          POPQUE: DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
```

```
042430  006301                          ASL     R1
042432  017102  041766                  MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
042436  005071  041766                  CLR     @QOUTPT(R1)     ;REMOVE DPB ADDRESS FROM THE QUEUE
042442  062761  000002  041766          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
042450  026161  041766  042010          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
042456  001003                          BNE     1$              ;NO--BRANCH TO EXIT
042460  016161  042006  041766          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
042466  006201                  1$:     ASR     R1
042470  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
                000122          $FIRST  =       MAXCYL+14       ;FIRST OPERATION INDICATOR

                                ;BAD SECTOR/TRACK ADDRESS STORAGE AREA INDEX EQUATE

                000124          $BDSEC  =       MAXCYL+16       ;BAD SECTOR STORAGE TABLE

                                ;DRIVE ID AREA INDEX EQUATE

                000224          $DRVID  =       $BDSEC+100      ;DRIVE ID

                                ;RH11/RM03 REGISTER EQUATES

                000234          $RMCS1  =       $DRVID+10       ;RM03 REGISTER STORAGE
                000236          $RMWC   =       $RMCS1+2
                000240          $RMBA   =       $RMCS1+4
                000242          $RMDA   =       $RMCS1+6
                000244          $RMCS2  =       $RMCS1+10
                000246          $RMDS   =       $RMCS1+12
                000250          $RMER1  =       $RMCS1+14
                000252          $RMAS   =       $RMCS1+16
                000254          $RMLA   =       $RMCS1+20
                000256          $RMDB   =       $RMCS1+22
                000260          $RMMR1  =       $RMCS1+24
                000262          $RMDT   =       $RMCS1+26
                000264          $RMSN   =       $RMCS1+30
                000266          $RMOF   =       $RMCS1+32
                000270          $RMDC   =       $RMCS1+34
                000272          $RMHR   =       $RMCS1+36
                000274          $RMMR2  =       $RMCS1+40
                000276          $RMER2  =       $RMCS1+42
                000300          $RMEC1  =       $RMCS1+44
                000302          $RMEC2  =       $RMCS1+46


                                ;BLOCK FOR DRIVE 0

042472     000      000         DRIVE0: .BYTE   0,0             ;DRIVE NUMBER
042474  000005                          .BLKW   5
042506  042726                          .WORD   .+$RMCS1-$REG
042510  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 1
```

```
042776     001     000          DRIVE1: .BYTE   1,0              ;DRIVE NUMBER
043000  000005                          .BLKW   5
043012  043232                          .WORD   .+$RMCS1-$REG
043014  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 2

043302     002     000          DRIVE2: .BYTE   2,0              ;DRIVE NUMBER
043304  000005                          .BLKW   5
043316  043536                          .WORD   .+$RMCS1-$REG
043320  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 3

043606     003     000          DRIVE3: .BYTE   3,0              ;DRIVE NUMBER
043610  000005                          .BLKW   5
043622  044042                          .WORD   .+$RMCS1-$REG
043624  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 4

044112     004     000          DRIVE4: .BYTE   4,0              ;DRIVE NUMBER
044114  000005                          .BLKW   5
044126  044346                          .WORD   .+$RMCS1-$REG
044130  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 5

044416     005     000          DRIVE5: .BYTE   5,0              ;DRIVE NUMBER
044420  000005                          .BLKW   5
044432  044652                          .WORD   .+$RMCS1-$REG
044434  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 6

044722     006     000          DRIVE6: .BYTE   6,0              ;DRIVE NUMBER
044724  000005                          .BLKW   5
044736  045156                          .WORD   .+$RMCS1-$REG
044740  000266                          .BLKB   $RMEC2-$REG


                                ;BLOCK FOR DRIVE 7

045226     007     000          DRIVE7: .BYTE   7,0              ;DRIVE NUMBER
045230  000005                          .BLKW   5
045242  045462                          .WORD   .+$RMCS1-$REG
045244  000266                          .BLKB   $RMEC2-$REG


                                ;GENERAL PURPOSE DPB - USED BY 'READHD', 'RECALT', 'OFFSET',&'RTNCTR'
```

```
045532  000000  000000  177776  GENDPB: .WORD   0,0,-2,CYLDER
045540  056150
045542  000000  000000  045552          .WORD   0,0,GENREG,0
045550  000000
045552  000024                  GENREG: .BLKW   24              ;REGISTER STORAGE IF ERROR

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

                                .SBTTL  ERROR MESSAGES

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

                                .NLIST  BEX

045622  044122  030067  030450  EM1:    .ASCIZ  /RH70(11) INTERRUPT OCCURRED (RMAS = 0)/
045671     125  042516  050130  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
045727     115  051501  041123  EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
045765     115  051501  041123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
046022  042101  051104  051505  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
046056  044122  030067  030450  EM6:    .ASCIZ  /RH70(11) DIDN'T RESPOND TO ADDRESSING/
046124  047125  047503  051122  EM10:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
046167     106  052101  046101  EM11:   .ASCIZ  /FATAL MASSBUS PARITY ERROR/
046222  042520  051522  051511  EM12:   .ASCIZ  /PERSISTENT DEVICE UNSAFE/
046253     117  042520  040522  EM13:   .ASCIZ  /OPERATION NOT COMPLETED WITHIN TIME LIMIT/
046325     104  044522  042526  EM14:   .ASCIZ  /DRIVE WENT OFFLINE/
046350  047516  051040  051505  EM15:   .ASCIZ  /NO RESPONSE TO PORT REQUEST/
046404  042510  042101  051105  EM20:   .ASCIZ  /HEADER CRC ERROR/
046425     104  052101  020101  EM21:   .ASCIZ  /DATA CHECK ('DCK') ERROR/
046456  051127  052111  020105  EM22:   .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') SET/
046531     127  044522  042524  EM23:   .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') NOT SET/
046610  042510  042101  051105  EM24:   .ASCIZ  /HEADER READ ERROR - 'FMT' BIT DROPPED/
046656  042510  042101  051105  EM25:   .ASCIZ  /HEADER READ ERROR - HEADER COMPARE ('HCE') ERROR/
046737     106  051117  040515  EM26:   .ASCIZ  /FORMAT ERROR ('FER')/
046764  042510  042101  051105  EM27:   .ASCIZ  /HEADER COMPARE ('HCE') ERROR/
047021     115  051511  042503  EM30:   .ASCIZ  /MISCELLANEOUS DRIVE ERROR/
```

```
047053     117  042520  040522  EM31:   .ASCIZ  /OPERATION INCOMPLETE ('OPI') ERROR/
047116  051104  053111  020105  EM32:   .ASCIZ  /DRIVE TIMING ('DTE') ERROR/
047151     120  051101  052111  EM33:   .ASCIZ  /PARITY ('PAR') ERROR AFTER OPERATION STARTED/
047226  051127  052111  020105  EM34:   .ASCIZ  /WRITE CLOCK FAILURE ('WCF') ERROR/
047270  047111  040526  044514  EM35:   .ASCIZ  /INVALID ADDRESS ('IAE') ERROR/
047326  051127  052111  020105  EM36:   .ASCIZ  /WRITE LOCK ('WLE') ERROR/
047357     104  052101  020101  EM37:   .ASCIZ  /DATA CHECK ('DCK') SET DURING WRITE CHECK COMMAND/
047441     122  030510  020061  EM40:   .ASCIZ  /RH11 OR UNIBUS TRANSFER ERROR/
047477     102  051525  040440  EM41:   .ASCIZ  /BUS ADDRESS OR WORD COUNT INCORRECT/
047543     104  052101  020101  EM42:   .ASCIZ  /DATA COMPARE ERRORS - NO OTHER ERROR(S) DETECTED/
047624  040503  023516  020124  EM43:   .ASCIZ  /CAN'T MATCH DATA READ WITH A PATTERN/
047671     105  051122  051117  EM44:   .ASCIZ  /ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11/
047755     105  041503  046040  EM45:   .ASCIZ  /ECC LOGIC FAILURE - POSITION REGISTER VALUE NOT VALID/
050043     102  051525  040440  EM46:   .ASCIZ  /BUS ADDRESS AND WORD COUNT NOT CONSISTENT/
050115     123  042505  020113  EM50:   .ASCIZ  /SEEK INCOMPLETE ('SKI') ERROR/
050153     120  047522  051107  EM51:   .ASCIZ  /PROGRAM DETECTED POSITIONING ERROR/
050216  051104  053111  020105  EM60:   .ASCIZ  /DRIVE UNSAFE ERROR/
050241     122  040515  000123  DH1:    .ASCIZ  /RMAS/
050246  051104  053111  020105  DH2:    .ASCIZ  /DRIVE   RMDS    RMER1   RMER2  RMMR2   RMAS/
050322  051104  053111  020105  DH3:    .ASCIZ  /DRIVE   REG ADR  DATA/
050350  051104  053111  020105  DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD    BAD/
050407     044  046522  042101  DH6:    .ASCIZ  /$RMADR/
050416  051104  020126  046522  DH14:   .ASCII  /DRV RMCS1   RMCS2   RMDS    RMER1   /
050462  046522  051115  020062          .ASCIZ  /RMMR2  RMER2   RMEC1   RMEC2/<CR><LF>
050521     122  053515  020103  DH15:   .ASCII  /RMWC    RMBA    RMDA    RMAS    RMLA    /
050571     122  042115  020102          .ASCIZ  /RMDB    RMMR1   RMDT/<CR><LF>
050620  046522  047123  020040  DH16:   .ASCIZ  /RMSN    RMOF    RMDC    RMMR2   STATUS/<CR><LF>
        050672                          .EVEN
                                .LIST   BEX
```

```
050672  001342  000000          DT1:     .WORD   ATTN,0

050676  001216  034150  034152  DT2:     .WORD   DRIVE,RMERRS,RMERRS+2,RMERRS+4,RMERRS+6,ATTN,0
050704  034154  034156  001342
050712  000000

050714  001216  041230  041232  DT3:     .WORD   DRIVE,RD.ADR,RD.WRD,0
050722  000000

050724  001216  041450  041446  DT4:     .WORD   DRIVE,WRT.AD,WRT.WD,RD.WRD,0
050732  041232  000000

050736  001270  000000          DT6:     .WORD   $RMADR,0

050742  000234  000244  000246  DT14:    .WORD   $RMCS1,$RMCS2,$RMDS,$RMER1,$RMMR2,$RMER2,$RMEC1,$RMEC2,0
050750  000250  000274  000276
050756  000300  000302  000000

050764  000236  000240  000242  DT15:    .WORD   $RMWC,$RMBA,$RMDA,$RMAS,$RMLA,$RMDB,$RMMR1,$RMDT,0
050772  000252  000254  000256
051000  000260  000262  000000

051006  000264  000266  000270  DT16:    .WORD   $RMSN,$RMOF,$RMDC,$RMMR2,$STATUS,0
051014  000274  000016  000000


                                .NLIST  BEX

051022  051120  051505  047105  LIN2C:   .ASCIZ  /PRESENT ORDER = /
051043     040  050040  042522  LIN2P:   .ASCIZ  /  PREVIOUS ORDER = /
051067     052  042440  051122  LIN2S:   .ASCIZ  @* ERROR AT BAD TRACK/SECTOR@
051123     105  051122  051117  LINM3:   .ASCIZ  /ERROR AT C/
051136  052040     000          T:       .ASCIZ  / T/
051141     120  042522  042523  LINN3:   .ASCIZ  /PRESENT ADDR = C/
051162  051440     000          S:       .ASCIZ  / S/
051165     040  020040  051120  LINP3:   .ASCIZ  /   PREV ADDR = C/
051206  052123  051101  020124  LINS3:   .ASCIZ  /START CYL = /
051223     040  042440  042116  LINEN3:  .ASCIZ  /  END CYL = /
051240  020040  041501  052524  LINA3:   .ASCIZ  /  ACTUAL CYL = /
051260  020040  051124  020113  LINT3:   .ASCIZ  /  TRK = /
051271     040  046522  041504  LINCA3:  .ASCIZ  / RMDC = /
051302  046522  040504  036440  LINDA3:  .ASCIZ  /RMDA = /
051312  046522  040502  036440  LINB3:   .ASCIZ  /RMBA = /
051322  020040  046522  041527  LINW3:   .ASCIZ  /  RMWC = /
051334  052123  051101  020124  LINST3:  .ASCIZ  /START TRK = /
051351     123  040524  052122  LINSS3:  .ASCIZ  /START SEC = /
051366  052502  043106  051105  LINM4:   .ASCIZ  /BUFFER ADDR = /
051405     040  051440  055111  LINS4:   .ASCIZ  /  SIZE = /
051417     040  040440  052103  LINX4:   .ASCIZ  /  ACTUAL NMBR WRDS XFRD = /
051452  047507  042117  042040  LIND5:   .ASCIZ  /GOOD DATA = /
051467     040  041040  042101  LINB5:   .ASCIZ  /  BAD DATA = /
051505     040  051440  041505  LINP5:   .ASCIZ  /  SECT POS = /
051523     110  040505  042504  LINS5:   .ASCIZ  /HEADER FROM ERROR SECTOR = /
051557     122  042515  030503  LINEP5:  .ASCIZ  /RMEC1 = /
051570  051040  042515  031103  LINEO5:  .ASCIZ  / RMEC2 = /
```

```
051602  042523  052103  051117  LINB6:   .ASCIZ  /SECTOR IS ECC CORRECTABLE /
051635     123  041505  047524  LINC6:   .ASCIZ  /SECTOR READ CORRECTLY /
051664  047503  051122  041505  LING6:   .ASCIZ  /CORRECTED ON /
051702  051040  052105  044522  LINR6:   .ASCIZ  / RETRIES/
051713     125  041516  051117  LINU06:  .ASCIZ  /UNCORRECTABLE AFTER /
051740  020040  047524  040524  LIN7M:   .ASCIZ  /  TOTAL MISPOS ERR = /
051766  051117  042504  051522  LIN7O:   .ASCIZ  /ORDERS:/
051776  052040  052117  046101  LIN7P:   .ASCIZ  / TOTAL SEEKS = /
052016  052040  052117  046101  LIN7S:   .ASCIZ  / TOTAL SKI ERR = /
052040  020040  051105  047522  LIN7T:   .ASCIZ  /  ERRORS:/
052052  020040  051127  051504  LIN7X:   .ASCIZ  /  WRDS XFR:/
052066  020040  051127  051504  LIN7R:   .ASCIZ  /  WRDS READ:/
052103     105  051122  051117  LIN8M:   .ASCIZ  /ERROR DURING RETRY/
052126  040504  040524  041440  LIN9B:   .ASCIZ  /DATA COMPARISON ERRORS/
052155     040  020040  020040  LIN9H:   .ASCII  /          GOOD    BAD/<CR><LF>
052203     114  041517  020040          .ASCIZ  /LOC       DATA    DATA/<CR><LF>
052233     114  041517  020040  LIN9I:   .ASCIZ  /LOC       DATA/<CR><LF>
052253     124  052117  046101  LIN9E:   .ASCIZ  /TOTAL COMPARE ERRORS = /
052303     124  042510  042040  LIN9G:   .ASCIZ  /THE DATA COMPARED OK/<CR><LF>
052332  051105  047522  020122  LIN10A:  .ASCIZ  /ERROR BURST BEGINS AT WORD /
052366  044440  020116  040504  LIN10B:  .ASCIZ  / IN DATA FIELD OF ERROR SECTOR/<CR><LF>
052427     105  051122  051117  LIN10C:  .ASCII  /ERROR WAS NOT IN THE DATA READ - /<CR><LF>
052472  041505  020103  047503          .ASCIZ  /ECC CORRECTION CAN'T BE PERFORMED/
052534  041505  020103  047503  LIN10H:  .ASCII  /ECC CORRECTION RESULTS/<CR><LF>
052564  042101  051104  020040          .ASCIZ  /ADDR    BAD     CORRECTED /<CR><LF>
052621     103  047117  042524  LIN11H:  .ASCIZ  /CONTENTS OF ERROR SECTOR (REPORTED ABOVE)/<CR><LF>
052675     101  042104  020122          .ASCIZ  /ADDR    DATA/<CR><LF>
052714  020040                  LIN4SP:  .ASCII  /  /
052716     040                  LINSP:   .ASCII  / /
052717     040     000          LINSPO:  .ASCIZ  / /
052721     122  052105  044522  LINX5:  .ASCIZ   /RETRIEVED BY A RDHD COMMAND/

                                .SBTTL  TELETYPE MESSAGES

052755     104  044522  042526  UNTMSG:  .ASCIZ  /DRIVE/
052763     040  043117  046106  UNTOFF:  .ASCIZ  / OFFLINE/
052774  047440  046116  047111  UNTON:   .ASCIZ  / ONLINE/
053004  047040  052117  041040  UNTNOT:  .ASCIZ  / NOT BEING TESTED/
053026  040440  051114  040505  UNTASN:  .ASCIZ  / ALREADY BEING TESTED/
053054  047040  052117  040440  NOTRM:   .ASCIZ  @ NOT AN PM03@
053071     040  047516  020124  NOTPRS:  .ASCIZ  / NOT PRESENT/
053106  047040  052117  040440  NOTAVL:  .ASCIZ  / NOT AVAILABLE/
053125     040  047125  040523  NOTSAF:  .ASCIZ  / UNSAFE/
053135     125  044516  020124  SYSTAT:  .ASCIZ  /UNIT STATUS:/<CR><LF><LF>
053155     122  030115  000063  RM03A:   .ASCIZ  /RM03/
053162  051104  053111  020105  STATHD:  .ASCII  /DRIVE PERFORMANCE SUMMARY/<CR><LF>
053215     104  053122  050040          .ASCII  /DRV PASS ORDERS   SEEKS   WRDS XFER   WRDS READ /
053275     123  043117  020124          .ASCIZ  /SOFT HARD  SKI MISP OTHER/<CR><LF>
053331     104  047117  006505  PDONE:   .ASCIZ  /DONE/<CR><LF><LF>
053341     007  043077  052101  DROPNG:  .ASCIZ  <07>/?FATAL OR EXCESSIVE ERRORS/
053375     105  042116  047440  ENDPAS:  .ASCIZ  /END OF PASS/
053411     015  042412  042116  ENDTST:  .ASCIZ  <CR><LF>/END OF TEST/
053427     015  042012  044522  DEASSG:  .ASCIZ  <CR><LF>/DRIVE DEASSIGNED/
053452  005015  025052  025052  DRNUM:   .ASCIZ  <CR><LF>/********** DRIVE #/
053477     040  052123  051101  ASGND:   .ASCIZ  / STARTED/<CR><LF>
053512  005015  020077  046047  NEDCLK:  .ASCIZ  <CR><LF>/? 'L' OR 'P' CLOCK REQUIRED ON SYSTEM/<CR><LF>
```

```
053564  000056                  PERIOD: .ASCIZ  /./
053566  000077                  QUES:   .ASCIZ  /?/
053570  047111  040526  044514  INVLD:  .ASCIZ  /INVALID COMMAND/<CR><LF>
053612  005015  047105  042524  ENTCOM: .ASCIZ  <CR><LF>/ENTER COMMANDS: /<CR><LF>
053637     105  052116  051105  ENTDRV: .ASCIZ  /ENTER I.D. FOR DRV #/
053664  005015  047105  042524  ENTLMT: .ASCIZ  <CR><LF>/ENTER ADDRESS LIMITS FOR DRV #/
053725     105  052116  051105  ENTADR: .ASCIZ  @ENTER BAD TRK/SEC ADRS FOR DRV #@
053766  000072                  COLON:  .ASCIZ  /:/
053770  005015  040504  042524  DATEIS: .ASCIZ  <CR><LF>/DATE: /
054001     015  047412  042520  IDIS:   .ASCIZ  <CR><LF>/OPERATOR I.D.: /
054023     015  005012  051104  HEDLIN: .ASCIZ  <CR><LF><LF>/DRV  DRV I.D./<CR><LF>
054046  047516  042516  005015  NONE:   .ASCIZ  /NONE/<CR><LF>
054055     077  044440  053116  BADENT: .ASCIZ  /? INVALID ENTRY/<CR><LF>
054077     123  051531  042524  BUSY:   .ASCIZ  /SYSTEM BUSY.../<CR><LF>
054120  005015  051120  043517  INTDON: .ASCII  <CR><LF>/PROGRAM INITIALIZATION COMPLETE/
054161     015  052012  050131          .ASCIZ  <CR><LF>/TYPE A 'CONTROL C' TO ENTER COMMANDS/<CR><LF><LF>
054233     015  043012  044501  MERR1:  .ASCIZ  <CR><LF>/FAIL TO RETRIEVE BAD SPOT FILE ON  /
054301     015  047412  042526  MERR2:  .ASCIZ  <CR><LF>/OVER 16 BAD SECTORS ARE DETECTED ON  /

        054352                  .EVEN

                                .LIST   BEX

                                ;PARAMETER ENTRY TABLE

054352  054500  020000  001462  PARLST: .WORD   PAR1,8192.,MAXDL
054360  054510  177777  001466         .WORD   PAR2,-1,INTRVL
054366  054660  177777  001460         .WORD   PAR19,-1,PASCNT
054374  054520  000017  001510         .WORD   PAR3,15.,PATTEN
054402  054620  000001  001476         .WORD   PAR11,1,WCSEL
054410  054630  000007  001500         .WORD   PAR14,7,RATIO
054416  054650  000001  001506         .WORD   PAR16,1,ENDET
054424  054610  000001  001474         .WORD   PAR10,1,FORMAT
                                ;      .WORD   PAR15,1,AUTOCK
054432  054670  000001  001504         .WORD   PAR20,1,NOTPRT
054440  054700  000001  001512         .WORD   PAR21,1,HEADER   ;RANDOM ADDRESS FLAG
054446  000000                         .WORD   0                ;TABLE TERMINATOR


                                .NLIST  BEX

054450  047105  042524  020122  ASKPAR: .ASCIZ  /ENTER PARAMETERS:  /
054474  027440  000040          SLASH:  .ASCIZ  @ / @

054500  044523  042532  020040  PAR1:   .ASCIZ  /SIZE   /
054510  044515  052516  042524  PAR2:   .ASCIZ  /MINUTE /
054520  040520  052124  051105  PAR3:   .ASCIZ  /PATTERN/
054530  040515  041530  046131  PAR4:   .ASCIZ  /MAXCYL /
054540  044515  041516  046131  PAR5:   .ASCIZ  /MINCYL /
054550  040515  052130  045522  PAR6:   .ASCIZ  /MAXTRK /
054560  044515  052116  045522  PAR7:   .ASCIZ  /MINTRK /
054570  040515  051530  041505  PAR8:   .ASCIZ  /MAXSEC /
054600  044515  051516  041505  PAR9:   .ASCIZ  /MINSEC /
054610  047506  046522  052101  PAR10:  .ASCIZ  /FORMAT /
054620  040522  042116  046517  PAR11:  .ASCIZ  /RANDOM /
```

```
054630  040522  044524  020117  PAR14:  .ASCIZ  /RATIO  /
054640  044103  041505  020113  PAR15:  .ASCIZ  /CHECK  /
054650  047105  044504  043516  PAR16:  .ASCIZ  /ENDING /
054660  040520  051523  051505  PAR19:  .ASCIZ  /PASSES /
054670  042515  051523  043501  PAR20:  .ASCIZ  /MESSAGE/
054700  042510  042101  051105  PAR21:  .ASCIZ  /HEADER /

                                        .EVEN


                                .LIST   BEX

                                ;PARAMETER TABLE POINTERS FOR ADDRESS LIMITS

054710  054730                  TABLE:  .WORD   TABLE0          ;PARAMETER TABLE FOR DRIVE 0
054712  054776                          .WORD   TABLE1          ;PARAMETER TABLE FOR DRIVE 1
054714  055044                          .WORD   TABLE2          ;PARAMETER TABLE FOR DRIVE 2
054716  055112                          .WORD   TABLE3          ;PARAMETER TABLE FOR DRIVE 3
054720  055160                          .WORD   TABLE4          ;PARAMETER TABLE FOR DRIVE 4
054722  055226                          .WORD   TABLE5          ;PARAMETER TABLE FOR DRIVE 5
054724  055274                          .WORD   TABLE6          ;PARAMETER TABLE FOR DRIVE 6
054726  055342                          .WORD   TABLE7          ;PARAMETER TABLE FOR DRIVE 7

                                ;PARAMETER TABLE FOR ADDRESS LIMITS

054730  054540  000000  042602  TABLE0: .WORD   PAR5,0,MINCYL+DRIVE0
054736  054530  000000  042600          .WORD   PAR4,0,MAXCYL+DRIVE0
054744  054560  000004  042606          .WORD   PAR7,4.,MINTRK+DRIVE0
054752  054550  000004  042604          .WORD   PAR6,4.,MAXTRK+DRIVE0
054760  054600  000037  042612          .WORD   PAR9,31.,MINSEC+DRIVE0
054766  054570  000037  042610          .WORD   PAR8,31.,MAXSEC+DRIVE0,0
054774  000000

054776  054540  000000  043106  TABLE1: .WORD   PAR5,0,MINCYL+DRIVE1
055004  054530  000000  043104          .WORD   PAR4,0,MAXCYL+DRIVE1
055012  054560  000004  043112          .WORD   PAR7,4.,MINTRK+DRIVE1
055020  054550  000004  043110          .WORD   PAR6,4.,MAXTRK+DRIVE1
055026  054600  000037  043116          .WORD   PAR9,31.,MINSEC+DRIVE1
055034  054570  000037  043114          .WORD   PAR8,31.,MAXSEC+DRIVE1,0
055042  000000

055044  054540  000000  043412  TABLE2: .WORD   PAR5,0,MINCYL+DRIVE2
055052  054530  000000  043410          .WORD   PAR4,0,MAXCYL+DRIVE2
055060  054560  000004  043416          .WORD   PAR7,4.,MINTRK+DRIVE2
055066  054550  000004  043414          .WORD   PAR6,4.,MAXTRK+DRIVE2
055074  054600  000037  043422          .WORD   PAR9,31.,MINSEC+DRIVE2
055102  054570  000037  043420          .WORD   PAR8,31.,MAXSEC+DRIVE2,0
055110  000000

055112  054540  000000  043716  TABLE3: .WORD   PAR5,0,MINCYL+DRIVE3
055120  054530  000000  043714          .WORD   PAR4,0,MAXCYL+DRIVE3
055126  054560  000004  043722          .WORD   PAR7,4.,MINTRK+DRIVE3
055134  054550  000004  043720          .WORD   PAR6,4.,MAXTRK+DRIVE3
055142  054600  000037  043726          .WORD   PAR9,31.,MINSEC+DRIVE3
055150  054570  000037  043724          .WORD   PAR8,31.,MAXSEC+DRIVE3,0
```

```
055156  000000

055160  054540  000000  044222  TABLE4: .WORD   PAR5,0,MINCYL+DRIVE4
055166  054530  000000  044220          .WORD   PAR4,0,MAXCYL+DRIVE4
055174  054560  000004  044226          .WORD   PAR7,4.,MINTRK+DRIVE4
055202  054550  000004  044224          .WORD   PAR6,4.,MAXTRK+DRIVE4
055210  054600  000037  044232          .WORD   PAR9,31.,MINSEC+DRIVE4
055216  054570  000037  044230          .WORD   PAR8,31.,MAXSEC+DRIVE4,0
055224  000000

055226  054540  000000  044526  TABLE5: .WORD   PAR5,0,MINCYL+DRIVE5
055234  054530  000000  044524          .WORD   PAR4,0,MAXCYL+DRIVE5
055242  054560  000004  044532          .WORD   PAR7,4.,MINTRK+DRIVE5
055250  054550  000004  044530          .WORD   PAR6,4.,MAXTRK+DRIVE5
055256  054600  000037  044536          .WORD   PAR9,31.,MINSEC+DRIVE5
055264  054570  000037  044534          .WORD   PAR8,31.,MAXSEC+DRIVE5,0
055272  000000

055274  054540  000000  045032  TABLE6: .WORD   PAR5,0,MINCYL+DRIVE6
055302  054530  000000  045030          .WORD   PAR4,0,MAXCYL+DRIVE6
055310  054560  000004  045036          .WORD   PAR7,4.,MINTRK+DRIVE6
055316  054550  000004  045034          .WORD   PAR6,4.,MAXTRK+DRIVE6
055324  054600  000037  045042          .WORD   PAR9,31.,MINSEC+DRIVE6
055332  054570  000037  045040          .WORD   PAR8,31.,MAXSEC+DRIVE6,0
055340  000000

055342  054540  000000  045336  TABLE7: .WORD   PAR5,0,MINCYL+DRIVE7
055350  054530  000000  045334          .WORD   PAR4,0,MAXCYL+DRIVE7
055356  054560  000004  045342          .WORD   PAR7,4.,MINTRK+DRIVE7
055364  054550  000004  045340          .WORD   PAR6,4.,MAXTRK+DRIVE7
055372  054600  000037  045346          .WORD   PAR9,31.,MINSEC+DRIVE7
055400  054570  000037  045344          .WORD   PAR8,31.,MAXSEC+DRIVE7,0
055406  000000


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

                                ;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

055410  104401  055532          OPRDAT: TYPE    ,ENTDAT         ;'ENTER DATE'
055414  104411                          RDLIN                   ;READ THE ENTRY
055416  012605                          MOV     (SP)+,R5        ;PUT THE ENTRY ADDRESS INTO R5
055420  112537  001320                  MOVB    (R5)+,DATE      ;STORE THE DATE
055424  112537  001321                  MOVB    (R5)+,DATE+1    ;STORE THE DATE
055430  112537  001322                  MOVB    (R5)+,DATE+2    ;STORE THE DATE
055434  112537  001323                  MOVB    (R5)+,DATE+3    ;STORE THE DATE
055440  112537  001324                  MOVB    (R5)+,DATE+4    ;STORE THE DATE
055444  112537  001325                  MOVB    (R5)+,DATE+5    ;STORE THE DATE
055450  112537  001326                  MOVB    (R5)+,DATE+6    ;STORE THE DATE
055454  112537  001327                  MOVB    (R5)+,DATE+7    ;STORE THE DATE
```

```
055460  005037  001330                  CLR     DATE+8.          ;SET TERMINATOR
055464  104401  055551                  TYPE    ,ENTID           ;'ENTER OPERATOR I.D.'
055470  104411                          RDLIN                    ;READ THE ENTRY
055472  012605                          MOV     (SP)+,R5         ;ENTRY ADDRESS
055474  112537  001332                  MOVB    (R5)+,OPERID     ;STORE THE I.D.
055500  112537  001333                  MOVB    (R5)+,OPERID+1   ;STORE THE I.D.
055504  112537  001334                  MOVB    (R5)+,OPERID+2   ;STORE THE I.D.
055510  112537  001335                  MOVB    (R5)+,OPERID+3   ;STORE THE I.D.
055514  112537  001336                  MOVB    (R5)+,OPERID+4   ;STORE THE I.D.
055520  112537  001337                  MOVB    (R5)+,OPERID+5   ;STORE THE I.D.
055524  005037  001340                  CLR     OPERID+6         ;SET TERMINTOR
055530  000207                          RTS     PC               ;RETURN

055532  005015  047105  042524  ENTDAT: .ASCIZ  <CR><LF>/ENTER DATE: /
055540  020122  040504  042524
055546  020072     000
055551     105  052116  051105  ENTID:  .ASCIZ  /ENTER OPERATOR I.D.: /
055556  047440  042520  040522
055564  047524  020122  027111
055572  027104  020072     000
        055600                  .EVEN

                                .SBTTL  ROUTINE TO SIZE MEMORY

                                ;;*****************************************************************
                                ;*CALL:
                                ;*      JSR     PC,$SIZE
                                ;*      RETURN
                                ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION

055600  010046                  $SIZE:  MOV     R0,-(SP)         ;;SAVE R0 ON THE STACK
055602  010146                          MOV     R1,-(SP)         ;;SAVE R1 ON THE STACK
055604  013746  000004                  MOV     @#ERRVEC,-(SP)   ;;SAVE PRESENT ERROR VECTOR PS & PC
055610  013746  000006                  MOV     @#ERRVEC+2,-(SP)
055614  010600                          MOV     SP,R0            ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
055616  104400                          TRAP                     ;;PUSH OLD PSW AND PC ON STACK
055620  012637  000006                  MOV     (SP)+,@#ERRVEC+2         ;;SAVE THE PSW IN @#ERRVEC+2
055624  012737  055644  000004          MOV     #2$,@#ERRVEC     ;;SET FOR TIMEOUT
055632  012701  020000                  MOV     #20000,R1        ;;FIRST ADDRESS
055636  005711                  1$:     TST     (R1)             ;;TEST THIS ADDRESS
055640  005721                          TST     (R1)+            ;;STEP TO NEXT ADDRESS
055642  000775                          BR      1$               ;;TRY ANOTHER
055644  162701  000002          2$:     SUB     #2,R1            ;;DROP BACK
055650  010006                          MOV     R0,SP            ;;RESTORE THE STACK
055652  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
055656  012637  000004                  MOV     (SP)+,@#ERRVEC
055662  010137  055674                  MOV     R1,$LSTAD        ;;LAST ADDRESS
055666  012601                          MOV     (SP)+,R1         ;;RESTORE R1
055670  012600                          MOV     (SP)+,R0         ;;RESTORE R0
055672  000207                          RTS     PC
055674  000000                  $LSTAD: .WORD   0                ;;CONTAINS THE LAST ADDRESS

                                .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
                                ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
                                ;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
```

```
                                        ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
                                        ;REQUIRED.
                                        ;NOTE: THIS ROUTINE DESTROYS R0-R4
                                        ;CALL
                                        ;
                                        ;       JSR     PC,BUSADR
                                        ;       RETURN

055676  005737  001356          BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
055702  002036                          BGE     3$              ;NO--BRANCH
055704  005037  001356                  CLR     CHGADR          ;YES--CLEAR THE REQUEST FLAG
055710  104401  001201                  TYPE    ,$CRLF          ;TYPE A CR-LF
055714  012700  001270          1$:     MOV     #$RMADR,R0      ;FIRST ADDRESS
055720  104401  056016                  TYPE    ,MRMCS1         ;"RMCS1="
055724  011046                          MOV     (R0),-(SP)      ;PRESENT RMCS1 ADDRESS
055726  104402                          TYPOC                   ;TYPE IT
055730  104401  052716                  TYPE    ,LINSP          ;2 SPACES
055734  104411                          RDLIN                   ;GET THE ENTRY
055736  012601                          MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
055740  004537  056040                  JSR     R5,CK.NUM       ;ENTER AND STORE THE NEW ADDRESS
055744  000763                          BR      1$              ;ERROR EXIT
055746  012700  001272          2$:     MOV     #$RMVEC,R0      ;VECTOR ADDRESS
055752  104401  056027                  TYPE    ,MRHVEC         ;"RHVEC="
055756  011046                          MOV     (R0),-(SP)      ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
055760  104402                          TYPOC                   ;TYPE IT
055762  104401  052716                  TYPE    ,LINSP          ;2 SPACES
055766  104411                          RDLIN                   ;READ THE ENTRY
055770  012601                          MOV     (SP)+,R1        ;ASCII TEXT ADDRESS
055772  004537  056040                  JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
055776  000763                          BR      2$              ;ERROR EXIT
056000  012700  001270          3$:     MOV     #$RMADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
056004  012701  034316                  MOV     #RMADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
056010  012021                          MOV     (R0)+,(R1)+     ;BUS ADDRESS
056012  012021                          MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
056014  000207                          RTS     PC              ;RETURN

056016  046522  051503  020061  MRMCS1: .ASCIZ  @RMCS1 = @
056024  020075     000
056027     122  053110  041505  MRHVEC: .ASCIZ  @RHVEC = @
056034  036440  000040

                                        .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
056040  010246                  CK.NUM: MOV     R2,-(SP)        ;SAVE R2
056042  010346                          MOV     R3,-(SP)        ;SAVE R3
056044  010446                          MOV     R4,-(SP)        ;SAVE R4
056046  012703  000006                  MOV     #6,R3           ;MAX OCTAL DIGITS IN THE NUMBER
056052  005002                          CLR     R2              ;FINAL OCTAL VALUE
056054  112104                  1$:     MOVB    (R1)+,R4        ;GET CURRENT POINTED BYTE
056056  001424                          BEQ     3$              ;BRANCH,IF TERMINATOR DETECTED
```

```
056060  120427  000060                  CMPB    R4,#'0          ;SMALLER THAN ASCII-0 ?
056064  103425                          BLO     5$              ;YES,ERROR EXIT
056066  120427  000067                  CMPB    R4,#'7          ;LARGER THAN ASCII-7 ?
056072  101022                          BHI     5$              ;YES,ERROR EXIT
056074  006302                          ASL     R2              ;SHIFT LEFT
056076  103420                          BCS     5$              ;
056100  006302                          ASL     R2              ;ONE
056102  103416                          BCS     5$              ;
056104  006302                          ASL     R2              ;OCTAL DIGIT
056106  103414                          BCS     5$              ;ERROR IF CARRY BIT SET
056110  042704  177770                  BIC     #177770,R4      ;CHOP OFF HIGHER BITS
056114  060402                          ADD     R4,R2           ;APPENDING CURRENT DIGIT TO NUMBER
056116  005303                          DEC     R3              ;DECREMENT BYTE COUNT
056120  001401                          BEQ     2$              ;BRANCH,IF LAST BYTE
056122  000754                          BR      1$              ;LOOPING BACK
056124  112104                  2$:     MOVB    (R1)+,R4        ;CHECK TERMINATOR
056126  001004                          BNE     5$              ;ERROR EXIT
056130  005702                  3$:     TST     R2              ;FINAL VALUE = 0
056132  001401                          BEQ     4$              ;YES,THEN NOT REPLACE THE ORIGINAL VALUE
056134  010210                          MOV     R2,(R0)         ;REPLACE THE ORIGINAL VALUE
056136  005725                  4$:     TST     (R5)+           ;ADJUST FOR NORMAL RETURN
056140  012604                  5$:     MOV     (SP)+,R4        ;RESTORE R4
056142  012603                          MOV     (SP)+,R3        ;RESTORE R3
056144  012602                          MOV     (SP)+,R2        ;RESTORE R2
056146  000205                          RTS     R5              ;EXIT

056150  000400                  CYLDER: .BLKW   256.            ;ONE SECTOR WORD CTR MAX SIZE
        057150                  ENDPGM  =       .


                                .NLIST  BEX

057150  005015  040515  047111  TITLE:  .ASCII  <CR><LF>/MAINDEC-11-DZRMB/<CR><LF>
057174  046522  031460  050040          .ASCIZ  @RM03 PERFORMANCE EXERCISER @<CR><LF><LF>
057233     015  052012  020117  LOADRV: .ASCII  <CR><LF>/TO TEST DRIVE 0, REPLACE THE 'XXDP' PACK ON DRIVE 0/<CR><LF>
057322  044527  044124  040440          .ASCII  /WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40,/<CR><LF>
057400  047101  020104  042522          .ASCIZ  /AND RESTART THE PROGRAM/<CR><LF>
057432  005015  054523  052123  NOLOAD: .ASCIZ  <CR><LF>/SYSTEM HAS 16K MEMORY, 'XXDP' LOADER WILL BE OVERWRITTEN/<CR><L


                                .LIST   BEX

        000001                          .END
```

```
ABASE = 176700          2271#   2561    2602
ABNRML  026674          3652    6681#
ACDW1 = 000000          2561    2604
ACDW2 = 000000          2561    2605
ACK   = 000123          2456#
ACPUOP= 000000          2561    2576
ACTDRV  034234          8090#   8358*   8409*   8727*   8736*   8985
ACTSTR  034235          8096#   8987*   9001*
ADDW0 = 000000          2561
ADDW1 = 000000          2561
ADDW10= 000000          2561
ADDW11= 000000          2561
ADDW12= 000000          2561
ADDW13= 000000          2561
ADDW14= 000000          2561
ADDW15= 000000          2561
ADDW2 = 000000          2561
ADDW3 = 000000          2561
ADDW4 = 000000          2561
ADDW5 = 000000          2561
ADDW6 = 000000          2561
ADDW7 = 000000          2561
ADDW8 = 000000          2561
ADDW9 = 000000          2561
ADEVCT= 000000          2561    2567
ADEVM = 000000          2561    2603
AENV  = 000000          2561    2572
AENVM = 000000          2561    2573
AFATAL= 000000          2561    2564
AMADR1= 000000          2561    2589
AMADR2= 000000          2561    2593
AMADR3= 000000          2561    2596
AMADR4= 000000          2561    2599
AMAMS1= 000000          2561    2583
AMAMS2= 000000          2561    2591
AMAMS3= 000000          2561    2594
AMAMS4= 000000          2561    2597
AMSGAD= 000000          2561    2569
AMSGLG= 000000          2561    2570
AMSGTY= 000000          2561    2563
AMTYP1= 000000          2561    2584
AMTYP2= 000000          2561    2592
AMTYP3= 000000          2561    2595
AMTYP4= 000000          2561    2598
AOE   = 001000          2361#
APASS = 000000          2561    2566
APRIOR= 000000          2561
APTCSU= 000040          7340    7446#
APTENV= 000001          7248    7333    7402    7444#
APTSIZ= 000200          3312    7443#
APTSPO= 000100          7335    7404    7445#
ASGN0   053477          3588    9573#
ASGN1   024250          6210#
ASGN2   024324          3473    6209    6222#
ASGN3   024406          6219    6230    6237#
ASGN4   024520          6238    6260#
```

```
ASGN6   024522          6245    6251#
ASGN7   024534          6244    6263#
ASKPAR  054450          3407    9592#
ASNERR  026544          6221    6235    6262    6272    6298    6318    6638#
ASNLST  001544          2751#   3544    3595*   5902    5909    6212    6228    6237    6287    6289*   6311    6320    6334
                        6657*   6671    6753*   6756
ASNMSG  026564          6211*   6216*   6227*   6261*   6266*   6268*   6270*   6297*   6317*   6646#
ASSIGN  024242          6208#   6277    6360    6365    6369
ASWREG= 000000          2561    2574
ATA   = 100000          2348#
ATABIT  034304          3595    5909    6212    6228    6237    6287    6289    6311    6334    6657    6753    8162#   8325
                        8421    8527    8908    8927    8935    8936    8956    9042
ATESTN= 000000          2561    2565
ATTN    001342          2627#   7228*   9547    9549
AT0   = 000001          2374#
AT1   = 000002          2375#
AT2   = 000004          2376#
AT3   = 000010          2377#
AT4   = 000020          2378#
AT5   = 000040          2379#
AT6   = 000100          2380#
AT7   = 000200          2381#
AUNIT = 000000          2561    2568
AUSWR = 000000          2561    2575
AUTOCK  001502          2704#
AVAIL   001612          2761#   3547    3589    3593*   3605    3608    3631    3655
AVECT1= 000254          2272#   2561    2600
AVECT2= 000000          2561    2601
A16   = 000400          2284#
A17   = 001000          2285#
BADENT  054055          6621    9573#
BADSEC  001362          2636#   3651*   5377*   5449    6030    6042    6054    6066    6078
BAI   = 000010          2302#
BEGCOD  001516          2732#   6398    6399
BEGPAT  001514          2731#   6401
BEGSIZ  001520          2739#   6403    6404
BIT0  = 000001          2256#   6534    6563
BIT00 = 000001          2246#   2256    4353    4989    5110    6285
BIT01 = 000002          2245#   2255    3728    4856    8401    8649
BIT02 = 000004          2244#   2254    3455    4402    4985    5138    5145    6432    6440
BIT03 = 000010          2243#   2253    3821    4188    8867    9159
BIT04 = 000020          2242#   2252    3821    3833    8902
BIT05 = 000040          2241#   2251    4289    8252    8669    8882    9037
BIT06 = 000100          2240#   2250    3902    3903    3993    4008    4848    5620    6202    7107    8334    8392    8466
                        8767    9219
BIT07 = 000200          2239#   2249    3714    3836    4848    8334    8601    8748    8867    8882    8902    8931    9191
BIT08 = 000400          2238#   2248    3830    8334
BIT09 = 001000          2237#   2247    3726    9035
BIT1  = 000002          2255#   6559    6573
BIT10 = 002000          2236#   3724    3848    3856    7236    8651
BIT11 = 004000          2235#   3722    3851    8299    8428    8620    8971
BIT12 = 010000          2234#   3720    3859    3902    3903    3922    4414    4992    5608    8294    8316    8334    8403
                        8436    8616    8635    8647    8660    8892    8894    9098    9154    9220
BIT13 = 020000          2233#   3839    3865    4160    4256    6933    7243    8386    8842    9101
BIT14 = 040000          2232#   3701    3703    3728    3824    3827    3865    3983    4034    4078    4118    4224    4239
                        8398    8433    8829    8967    9063    9109    9113    9217
```

```
BIT15 = 100000      2231#   3699    3867    3902    3962    4006    4256    8386    8398    8401    8403    8433    8436
                    8620    8649    8651    8767    8867    8882    8892    8902    8967    9035    9063    9070
BIT2  = 000004      2254#   3730    9070
BIT3  = 000010      2253#   3842
BIT4  = 000020      2252#   4084
BIT5  = 000040      2251#   3845
BIT6  = 000100      2250#   3922    5469
BIT7  = 000200      2249#   3915    4124    4822
BIT8  = 000400      2248#   4039    4063    4068    4098    4103    4256
BIT9  = 001000      2247#   3854    4256
BLKADR  002102      2807#   3360    3501    5907    6248    6254    6291    6314
BPTVEC= 000014      2263#
BUFTBL  001700      2773#   3379*   3380*   3381*   3384*   3385*   3387*   3388*   3391*   3395    3397    3398    4883
                    4885    4899*   4925    4926    4938*   4943*   4948    4949    4962*
BUSADR  055676      3351    9758#
BUSY    054077      6145    9573#
CFLAG   001360      2635#   3370*   3533    6148*   6154    6471*   6481    6598*   6610    6626*   7120*   7177*
CHGADR  001356      2634#   3272*   3274*   3471    3490    9758    9760*
CHKADR  016762      3686    5115    5161#
CI1     035570      8456    8484#
CI3     035676      8458    8507#
CI4     035776      8447    8529#
CI5     036354      8506    8528    8606#   8615
CI6     036376      8545    8555    8557    8559    8611#
CI7     036412      8388    8465    8496    8500    8504    8512    8522    8526    8537    8544    8550    8554    8568
                    8574    8578    8588    8599    8614    8616#   8722    8752    8919    9203
CI7B    036440      8619    8622#   8921
CI8     036512      8617    8625    8634#   8796    9055
CKBUS   012532      3706    4384#
CKCLK   022412      3421    5863#
CKCLK1  022506      5865    5876#
CKCLK2  022554      5877    5885#
CKCLK3  022604      5875    5884    5892#
CKERR   012440      3705    4365#
CKFMT   010512      3835    4063#
CKHCE   010706      3838    4098#
CKSWR = 104407      7227    7258    7974#
CK.CHR  030036      6989#   7026    7040
CK.DEC  030010      6967#   6995
CK.DIG  030110      6612    7020#
CK.NUM  056040      9769    9778    9799#
CK.OCT  027762      6949#   6997
CLKFLG  001310      2618#   5863*   5868*   5879*   6088
CLOCK   023546      5871    5881    6109#
CLR   = 000040      2304#
CLRDPB  025212      3502    6249    6377#
CLRQUE  042230      8234    8668    9287#
CMCNT   001410      2651#   4352*   4356*   4411*   4412*   4422    4424    4425*   4428*   4477    4481    4484*   4506*
CMCYL   001412      2652#   4413*   4414*   4415*   4431    4496*   4497    4500
CMDAT   013064      4430    4439    4443#   4485    4487
CMHED   013016      4431#
CMPAR   012616      3709    4402#
CMPARD  012634      3951    4406#
CMPLMT  001472      2688#   4360    4417
CMPRES  017456      3566    3633    3659    5272#   5275
CMPRT   013426      4441    4455    4465    4475    4515#
```

```
CMPRX   013420          4438    4478    4483    4499    4507    4510#
CMSEC   001414          2653#   4416*   4434    4488*   4489    4491*   4503
CMSTR   012734          4419#
CMSTR2  013052          4433    4436    4440#   4502    4505
CMTRK   001415          2654#   4492*   4493    4495*
COLON   053766          6095    6100    9573#
COMTBL  002122          2816#   5238    6400
CR    = 000015          2171#   2608#   7203    7379    7389    9545    9573    9707    9837
CRLF  = 000200          2172#   7350    7389
CYLDER  056150          4105*   4106    4109*   4143*   5607    5662    5665    6539    9533    9832#
CYLIMT  001446          2667#   3507    5096*   5102    5106    6439*   6446    6454    6455
DATAPK  025150          6182    6359#
DATA0   002542          2913    2929    2932#
DATA1   002602          2914    2915    2916    2917    2918    2919    2920    2921    2922    2923    2924    2925    2926
                        2927    2928    2930    2949#
DATE    001320          2622#   6324    6327    9686*   9687*   9688*   9689*   9690*   9691*   9692*   9693*   9694*
DATEIS  053770          6326    9573#
DCK   = 100000          2367#
DCKER   007312          3864    3875#
DCKER1  007432          3898#   4178
DDISP = 177570          2178#   2544    3300
DEASGN  024606          6172    6281#
DEASSG  053427          3562    9573#
DH1     050241          3224    9545#
DH14    050416          5454    9545#
DH15    050521          5463    9545#
DH16    050620          5466    9545#
DH2     050246          3231    3252    9545#
DH3     050322          3238    9545#
DH4     050350          3245    9545#
DH6     050407          3259    9545#
DISPLA  001156          2544#   3300*   3308*   7235*
DISPLY= 104414          3746    3780    3800    3811    3883    3897    3918    3955    3965    3987    4029    4054    4074
                        4114    4140    4155    4177    4183    4201    4211    4220    4235    4251    4271    4282    4301
                        4319    4324    4325    4326    4329    4332    4335    4338    4341    4344    4347    4350    4374
                        4392    4539    4544    4545    4546    4559    4562    4565    4574    4577    4633    4638    4670
                        4673    4676    4679    4682    4685    4688    4691    4693    4694    4717    4723    4728    4730
                        4839    5398    5401    5418    5444    5446    5451    5452    5453    5454    5455    5458    5463
                        5466    5477    5478    5491    5494    5500    5506    5513    5527    5530    5533    5539    5542
                        5543    5547    5548    5552    5560    5563    5564    5567    5575    5578    5580    5583    5587
                        5591    5596    5599    5606    5610    5614    5622    5625    5628    5633    5639    5643    5652
                        5654    5661    5664    5667    5668    5669    5675    5678    5679    5682    5688    5689    5695
                        5701    5707    5713    5720    5722    5730    5731    5737    5742    5745    5750    5755    5756
                        5765    5770    5775    5778    5781    5782    5791    5805    6912    7979#
DISPRE  000174          2475#   3308
DLT   = 100000          2314#
DONE    007004          3716    3821#
DPE   = 020000          2433#
DPINT   034210          8050#   8263    8266    8342*   8368    8813    8838    8957    8959*   9026    9044    9049    9057*
DPR   = 000400          2341#
DPRQS   034220          8063#   8374    8422*   8460*   8816    9028    9046    9051    9066*
DRIVE = 001216          2[illegible]#   7229*   9549    9553    9556
DRIVE0  042472          2807    6333    9469#   9608    9609    9610    9611    9612    9613
DRIVE1  042776          2808    9477#   9616    9617    9618    9619    9620    9621
DRIVE2  043302          2809    9485#   9624    9625    9626    9627    9628    9629
DRIVE3  043606          2810    9493#   9632    9633    9634    9635    9636    9637
```

```
DRIVE4   044112          2811    9501#   9640    9641    9642    9643    9644    9645
DRIVE5   044416          2812    9509#   9648    9649    9650    9651    9652    9653
DRIVE6   044722          2813    9517#   9656    9657    9658    9659    9660    9661
DRIVE7   045226          2814    9525#   9664    9665    9666    9667    9668    9669
DRNUM    053452          3738    3764    3792    6663    6721    6729    9573#
DROP     026574          3752    3806    6655#   6685
DROPNG   053341          6662    9573#
DRQ    = 004000          2397#
DRVACT   034160          8009#   8382    8562*   8609*   8628*   8639    8654*   8741*   8834    8865    8896*   8903*   8916
                         8926*   9038*
DRVCLR= 000111           2451#
DRVER    010460          3869    4053#
DRVINT   034546          8254    8288#   8370    8947    8960
DRVPRM   025412          6250    6419#
DRVQUE   042326          8377    8390    8779    9327#
DRVSTA   J34170          3438    3499    6243    8023#   8244*   8245*   8246*   8247*   8257*   8289*   8298*   8331*   8336*
                         8364    8372    8396    8430    8434    8563*   8672*   8819    8827    8840    8890*   8897*   8909
                         8962    9058*
DRVTYP   034200          3441    3455    6263    6432    6440    6443    8037#   8290*   8305*   8310*   8399    8673*
DRY    = 000200          2340#
DSWR   = 177570          2177#   2543    3299
DTE    = 010000          2364#
DTEER    011316          3861    4174#
DTUW     034302          8156#   8240    8449    8505*   8629    8632*   8642    8655    8657*   8666*   8730    8742*   9031
                         9107
DT00   = 000001          2388#
DT01   = 000002          2389#
DT02   = 000004          2390#
DT03   = 000010          2391#
DT04   = 000020          2392#
DT05   = 000040          2393#
DT06   = 000100          2394#
DT07   = 000200          2395#
DT08   = 000400          2396#
DT1      050672          3225    9547#
DT14     050742          5459    9561#
DT15     050764          5464    9565#
DT16     051006          5467    9569#
DT2      050676          3232    3253    9549#
DT3      050714          3239    9553#
DT4      050724          3246    9556#
DT6      050736          3260    9559#
DULP     034550          8289#   8300
DUNIT    001546          2753#   3538    6291*   6659*   6755*
DVA    = 004000          2334#
DVC    = 000200          2429#
ECBAD0   001432          2661#   4651*   4674
ECBAD1   001440          2664#   4665*   4686
ECBIT    001416          2655#   4625*   4626*   4627    4628*   4645    4647*
ECC      014026          3944    4609#
ECCX     014560          4681    4692    4694#
ECC1     014424          4658    4664    4670#
ECC2     014554          4642    4693#
ECGD     001430          2660#
ECGD1    001436          2663#
ECH    = 000100          2358#
```

```
ECI    = 004000    2417#
ECMSK0   001422    2657#   4643*   4648*   4653    4655
ECMSK1   001424    2658#   4644*   4649*   4657    4666    4668
ECSEC    001420    2656#   4609*   4618*   4621*   4623*   4639
ECWRD    001426    2659#   4627*   4629*   4630*   4631*   4632*   4634    4639*   4640    4651    4654    4655*   4656*
                   4659    4671    4677
ECWRD1   001434    2662#   4652*   4659*   4660*   4661    4663*   4665    4667    4668*   4669*   4680    4683    4689
EMPTYQ   042306    8626    8671    8768    8904    8969    9071    9312#
EMTVEC=  000030    2266#   3287*   3288*   3316*
EM1      045622    3223    9545#
EM10     046124    3757    9545#
EM11     046167    3763    9545#
EM12     046222    3737    3746    9545#
EM13     046253    3780    9545#
EM14     046325    3791    3800    9545#
EM15     046350    3811    9545#
EM2      045671    3230    9545#
EM20     046404    4029    9545#
EM21     046425    3897    9545#
EM22     046456    3982    9545#
EM23     046531    3965    9545#
EM24     046610    4074    9545#
EM25     046656    4114    9545#
EM26     046737    4220    9545#
EM27     046764    4235    9545#
EM3      045727    3237    9545#
EM30     047021    4054    9545#
EM31     047053    4155    9545#
EM32     047116    4177    9545#
EM33     047151    4183    9545#
EM34     047226    4282    9545#
EM35     047270    4201    9545#
EM36     047326    4211    9545#
EM37     047357    3985    9545#
EM4      045765    3244    9545#
EM40     047441    4251    9545#
EM41     047477    4392    9545#
EM42     047543    4539    9545#
EM43     047624    4319    4324    9545#
EM44     047671    4374    9545#
EM45     047755    3883    9545#
EM46     050043    5477    9545#
EM5      046022    3251    9545#
EM50     050115    4271    9545#
EM51     050153    4140    9545#
EM6      046056    3258    9545#
EM60     050216    4301    9545#
ENDCMP   013702    4510    4570#
ENDCON   001450    2675#   3487*   3488*   6700    6703
ENDET    001506    2713#   6698    9584
ENDPAS   053375    6712    9573#
ENDPGM=  057150    3380    3385    9833#
ENDSEK   001454    2677#   6706    6709
ENDTST   053411    6720    9573#
ENTADR   053725    9573#
ENTCOM   053612    6151    9573#
```

```
ENTDAT  055532          9683    9707#
ENTDRV  053637          6472    9573#
ENTID   055551          9695    9710#
ENTLMT  053664          6423    9573#
ENTPR   004622          3412#
EOP     026722          3653    6698#
EOPX    027220          6702    6705    6708    6710    6749    6757    6761#
EOP1    026754          6699    6706#
EOP2    026776          5313    6701    6704    6707    6711#
ERCTR   001404          2649#   4409*   4440*   4451    4464*   4572    4575
ERPRC1  006364          3715    3720#   3919    4840
ERPROC  006350          3698    3702    3704    3714#
ERR   = 040000          2347#
ERRVEC= 000004          2259#   3297    3298*   3309*   5865*   5866*   5877*   5892*   9726    9727    9731*   9732*   9739#
                        9740#
FACTOR  015416          4850*   4851*   4853*   4854    4862    4870#
FAIRNS  001352          2632#
FALPAR  006550          3725    3762#
FALPR1  006560          3758    3764#
FER   = 000020          2356#
FILBUF  016004          3621    4984#
FISK    027274          6779    6780#
FMTER   011512          4070    4219#
FMT16 = 010000          2418#
FORMAT  001474          2689#   3676    5191    9585
FOWT1   005140          3484#
FOWT2   005200          3489    3492#
FOWT3   005342          3491    3520#
FRSTER  001376          2642#   3950*   3952    4316    4358*   4359*   4405*   4467    4517    4535    4547*   4570
F0    = 000002          2329#
F1    = 000004          2330#
F2    = 000010          2331#
F3    = 000020          2332#
F4    = 000040          2333#
GENDPB  045532          3928#   3932    4739*   4740*   4742    4759*   4760*   4762    4764    4775*   4776*   4778    4791*
                        4792*   4793*   4794*   4795*   4796*   4798    4800    6241*   6513*   6514*   6515*   6516*   6517*
                        6524*   6526    6528    6531*   6532    6561*   9533#
GENREG  045552          4068    4103    9535    9537#
GETADR  025764          3503    6252    6502#
GETBUF  015420          3618    4880#
GETID   025654          6251    6468#
GETPAR  017164          3616    5221#
GETPAT  017122          3689    5149    5201#   5210    5336
GETREG= 000141          2458#
GETREM  027304          5042    5071    5087    5104    5126    5206    6789#
GETREQ  042402          8423    8644    8759    8823    8857    8918    8966    9034    9060    9067    9348#
GNS   = ****** U        2474    7966    7967    7968    7969    7970    7972    7974    7975    7976    7977    7978    7979
GO    = 000001          2328#
GODRIV  016132          3623    3911    3996    4815    5018#
GTSWR = 104406          3334    7972#
HCE   = 000200          2359#
HCEER   011570          4110    4234#
HCI   = 002000          2416#
HCRC  = 000400          2360#
HCRCER  010314          3832    4024#   4065    4100
HEADER  001512          2720#   5051    9588
```

```
HEDLIN  054023          6332    9573#
HOUR    001364          2637#   3365*   6091    6121*   6122    6124*
HT    = 000011          2169#   7348    7389
HZ      001312          2619#   3364    6111
IAE   = 002000          2362#
IAEER   011432          3850    4200#
IDIS    054001          6330    9573#
IDLE    006006          3532    3603    3607    3641#
IDLE1   006104          3643    3661#
IE    = 000100          2282#
ILF   = 000001          2352#
ILR   = 000002          2353#
INCHRO  023330          3936    6042#
INCMIS  023400          4145    6066#
INCSKI  023354          4275    6054#
INCSOF  023304          3943    3949    4037    4082    4122    6030#
INCTOT  023424          3750    3771    3784    3804    3815    3885    3917    3937    3954    3971    4014    4017    4038
                        4057    4083    4123    4146    4159    4187    4204    4213    4228    4243    4255    4274    4286
                        4304    4378    4396    4578    6078#
INTDON  054120          3520    9573#
INTRVL  001466          2684#   3368*   6116*   6127    6129    6132*   9579
INVLD   053570          6199    9573#
IOTVEC= 000020          2264#
IR    = 000100          2305#
ISR     037122          8249    8727#
IVC   = 010000          2432#
KIPARO= ****** U        9722
KSR     023710          3535    6143#
KSR1    023722          6144    6146#
LA      036746          8453    8688#
LACNT   034246          8115#   8696*   8697    8718*
LBC   = 002000          2430#
LBT   = 002000          2343#
LF    = 000012          2170#   2609#   7203    7383    7389    9545    9573    9707    9837
LIMIT   001406          2650#   4360*   4417*   4418*   4549    4551*
LINA3   051240          5606    9573#
LINB3   051312          5527    9573#
LINB5   051467          5643    9573#
LINB6   051602          5688    5701    9573#
LINCA3  051271          5564    9573#
LINC6   051635          5695    9573#
LINDA3  051302          5560    9573#
LINDEC  022242          4576    5457    5541    5546    5551    5574    5577    5579    5582    5586    5590    5598    5601
                        5609    5613    5627    5632    5653    5729    5744    5777    5780    5817#
LINDS   051452          5639    9573#
LINEN3  051223          5599    9573#
LINEOS  051570          5679    9573#
LINEPS  051557          5675    9573#
LINE1   020176          3745    3779    3799    3810    3882    3896    3964    3981    4028    4053    4073    4113    4139
                        4154    4176    4182    4200    4210    4219    4234    4250    4270    4281    4300    4318    4373
                        4391    4538    5393#
LINE2   020242          3747    3781    3801    3812    3884    3898    3967    3989    4030    4055    4075    4115    4141
                        4156    4184    4202    4212    4221    4236    4252    4272    4283    4302    4320    4375    4393
                        4540    5414#   5792
LINE2A  020412          5426    5431    5449#
LINE2B  020430          5450    5453#
```

LINE3	020650	3748	3782	3802	3813	3899	3968	3990	4031	4056	4076	4116	4157	4222
		4237	4253	4303	4376	5500#								
LINE3A	020656	4284	4321	4541	5506#									
LINE3B	020664	4273	5512#											
LINE3C	020676	4142	5519#											
LINE3D	020706	4394	5479	5525#										
LINE3E	020754	4185	5539#											
LINE3F	021042	4203	5558#											
LINE4	021320	3749	3783	3803	3814	3900	3969	3991	4032	4077	4117	4158	4186	4223
		4238	4254	4285	4322	4377	4395	4542	5480	5620#				
LINE5	021410	3970	3992	4036	4080	4120	4226	4241	5639#					
LINE5A	021510	4033	4081	4121	4144	4227	4242	5660#						
LINE5B	021556	3942	5675#											
LINE6	021620	3905	5688#											
LINE6A	021632	3948	5695#											
LINE6B	021640	3941	5701#											
LINE6C	021646	3889	3975	4001	4010	4043	4088	4128	4164	4192	4262	4292	4308	5707#
LINE6D	021654	3891	3946	3977	4046	4091	4131	4167	4195	4264	4295	4310	5713#	
LINE7	021730	3751	3785	3805	3816	3892	3938	3956	4015	4018	4044	4047	4058	4089
		4092	4129	4132	4165	4168	4193	4205	4214	4229	4244	4265	4293	4311
		4379	4397	4579	4836	5737#								
LINE7A	022056	4147	4276	5765#										
LINE8	022176	3935	4012	4835	5791#									
LING6	051664	5707	9573#											
LINKDV	027332	4614	4706	5650	6792	6802#								
LINM3	051123	5500	9573#											
LINM4	051366	5622	9573#											
LINN3	051141	5506	9573#											
LINOCT	022210	4328	4331	4334	4337	4340	4343	4346	4349	4558	4561	4564	4672	4675
		4678	4684	4687	4690	4720	4725	5489	5529	5532	5562	5566	5624	5642
		5645	5663	5666	5677	5681	5801#							
LINP3	051165	5580	9573#											
LINP5	051505	5652	9573#											
LINR6	051702	5730	9573#											
LINSP	052716	3453	4329	4335	4341	4347	4559	4562	4673	4676	4679	4685	4688	4691
		4723	5458	5491	5542	5547	5563	5664	5667	5678	5962	5967	5972	5978
		5984	9573#	9766	9775									
LINSPO	052717	5401	5455	5990	5997	6002	6007	6012	9573#					
LINSS3	051351	5548	9573#											
LINST3	051334	5543	9573#											
LINS3	051206	5539	5596	9573#										
LINS4	051405	5625	9573#											
LINS5	051523	5661	9573#											
LINT3	051260	5610	9573#											
LINU06	051713	5713	9573#											
LINW3	051322	5530	9573#											
LINX4	051417	5628	9573#											
LINX5	052721	5668	9573#											
LIN10A	052332	4633	9573#											
LIN10B	052366	4638	9573#											
LIN10C	052427	4693	9573#											
LIN10H	052534	4670	9573#											
LIN11H	052621	4717	9573#											
LIN2C	051022	5421	9573#											
LIN2P	051043	5427	9573#											
LIN2S	051067	5452	9573#											

```
LIN3.1   021114          5501    5507    5573#
LIN3.3   021222          5512    5519    5596#
LIN3.4   021254          5520    5606#
LIN4SP   052714          3437    6341    9573#
LIN6.1   021662          5696    5702    5718#
LIN6.2   021704          5708    5714    5727#
LIN7M    051740          5775    9573#
LIN7O    051766          5737    5765    9573#
LIN7P    051776          5770    9573#
LIN7R    052066          5750    9573#
LIN7S    052016          5778    9573#
LIN7T    052040          5742    9573#
LIN7X    052052          5745    9573#
LIN8M    052103          3918    4839    5791    9573#
LIN9B    052126          4544    9573#
LIN9E    052253          4574    9573#
LIN9G    052303          3955    9573#
LIN9H    052155          4546    9573#
LIN9I    052233          4326    9573#
LKPAR    004564          3402#
LOADRV   057233          3325    9837#
LSC    = 004000          2431#
LSTAD    001354          2633#   3378*   3381    3382    3478
MAIN     005352          3530#   3668    5317    5343
MAINDA   005376          3537#
MAIN1    005524          3519    3521    3545    3571#   6781
MAIN2    005640          3581    3602#
MASK     001346          2629#   3902*   3920    3966*   4039*   4084*   4124*   4160*   4188*   4259*   4289*   4824    4829
MATCH    013750          4443    4590#
MAXCYL=  000106          3507*   5055    5096    5097    5302    6446*   9427#   9428    9429    9430    9431    9432    9433
                         9437    9609    9617    9625    9633    9641    9649    9657    9665
MAXDL    001462          2680#   3392    3394*   3395    3397*   3414    3416*   5122    5135    5137    5318    5320    9578
MAXER    001464          2683#   3371*   6683
MAXSEC=  000116          3509*   5064    6450*   9431#   9613    9621    9629    9637    9645    9653    9661    9669
MAXTRK=  000112          3508*   5080    5305    6448*   9429#   9611    9619    9627    9635    9643    9651    9659    9667
MCPE   = 020000          2287#
MCPEMX   034314          8174#   9088    9135
MDPE   = 000400          2307#
MERR1    054233          6565    9573#
MERR2    054301          6575    9573#
MINCYL=  000110          3510*   5057    5098    5100    5105    5108    5309    6447*   6461    9428#   9608    9616    9624
                         9632    9640    9648    9656    9664
MINSEC=  000120          3512*   5058    5065    5067    5072    5308    6394    6451*   6459    9432#   9612    9620    9628
                         9636    9644    9652    9660    9668
MINTRK=  000114          3511*   5059    5081    5083    5088    5307    6449*   6460    9430#   9610    9618    9626    9634
                         9642    9650    9658    9666
MINUTE   001366          2638#   3366*   6096    6117*   6118    6120*
MNDLTA   034330          8190#   8716
MNTBL    002152          2844#   5442
MOH    = 020000          2398#
MOL    = 010000          2345#
MONTR    005064          3465    3469#
MRHVEC   056027          9772    9788#
MRMCS1   056016          9763    9786#
MXDLTA   034326          8187#   8714
MXF    = 001000          2308#
```

```
MXLACT  034324          8184#   8697
MXWNDW  034332          8193#
M.DPID  027400          6809    6823#
M.DP40  027436          6834#   6855
M.DP41  027472          6839    6846#
M.DP42  027502          6845    6850#
M.DP44  027534          6857    6862#
M.DP50  027546          6832    6866#
NED   = 010000          2311#
NEDCLK  053512          5886    9573#
NEM   = 004000          2310#
NEWASN  024576          6168    6276#
NEWUNT  001570          2757#   3505*   3575    3593    3594*   6254*
NOLOAD  057432          3480    9837#
NOMTCH  012166          4316#   4445
NONE    054046          6344    9573#
NOTAVL  053106          6216    9573#
NOTPRS  053071          3446    6268    9573#
NOTPRT  001504          2708#   5375    9587
NOTRM   053054          3444    6266    9573#
NOTSAF  053125          3450    6261    9573#
NSA   = 100000          2400#
OFFCOD  002464          2896#   3928
OFFDIR= 000001          2415#
OFFON = 000001          2338#
OFFSET= 000115          2453#   4776
OFFST   015016          3931    4775#
OFLIN   006656          3729    3790#
OFMSG0  002362          2880#   2902
OFMSG1  002415          2885#   2903
OFMSG2  002440          2889#   2904
OFMTBL  002470          2902#   5719
OPE   = 000010          2428#
OPERID  001332          2624#   6328    6331    9698*   9699*   9700*   9701*   9702*   9703*   9704*
OPI   = 020000          2365#
OPIER   011206          3841    4152#
OPIER1  011252          4161#
OPRDAT  055410          3356    9683#
OPT     035332          8384    8419#   8762    8940    8977
OPTBL   002130          2823#   5433    5435
OR    = 000200          2306#
ORDERQ  001522          2747#   3358    3530    3602    3624    3641    6143
PACK    001316          2621#   3369*   3518*   5340*   6256    6258    6276*   6359*   6364*   6368*
PAR   = 000010          2355#
PARENT  026420          3413    6458    6597#
PARER   011340          3844    4182#
PARLST  054352          3398*   3412    9578#
PAR0    001656          2769#
PAR1    054500          9578    9592#
PAR10   054610          9585    9592#
PAR11   054620          9582    9592#
PAR14   054630          9583    9592#
PAR15   054640          9592#
PAR16   054650          9584    9592#
PAR19   054660          9580    9592#
PAR2    054510          9579    9592#
```

```
PAR20   054670          9587    9592#
PAR21   054700          9588    9592#
PAR3    054520          9581    9592#
PAR4    054530          9592#   9609    9617    9625    9633    9641    9649    9657    9665
PAR5    054540          9592#   9608    9616    9624    9632    9640    9648    9656    9664
PAR6    054550          9592#   9611    9619    9627    9635    9643    9651    9659    9667
PAR7    054560          9592#   9610    9618    9626    9634    9642    9650    9658    9666
PAR8    054570          9592#   9613    9621    9629    9637    9645    9653    9661    9669
PAR9    054600          9592#   9612    9620    9628    9636    9644    9652    9660    9668
PASCNT  001460          2679#   6718    9580
PAT   = 000020          2303#
PATTEN  001510          2717#   5202    5204    9581
PCLOCK  001306          2617#   5864*   5869*
PDONE   053331          9573#
PERIOD  053564          9573#
PGE   = 002000          2309#
PGM   = 001000          2342#
PIP   = 020000          2346#
PIRQ  = 177772          2176#
PIRQVE= 000240          2270#
POPQUE  042424          8432    8484    8600    8930    9363#
POSER   011132          4108    4138#
PROCES  006260          3650    3696#
PRTBAD  014566          3901    3939    4287    4699#
PRTIM   006746          3731    3810#
PR0   = 000000          2193#
PR1   = 000040          2194#
PR2   = 000100          2195#
PR3     000140          2196#
PR4   = 000200          2197#   6147
PR5   = 000240          2198#
PR6   = 000300          2199#
PR7   = 000340          2200#
PS    = 177776          2173#   2174    3363*   3464*   6147*   6203*   6920    6921*   6935*   7129    8232    8233*   8262*
                        8270*   8356    8357*   8410*   8420    8469*   8703*
PSEL  = 002000          2286#
PSW   = 177776          2174#
PUNSAF  006446          3721    3736#
PWRVEC= 000024          2265#   3291*   3292*   7452*   7453*   7462*   7468*   7480*   7481*
QCNT    041736          9229#   9288    9312*   9327    9329*   9349    9363*
QDRV0   042030          9240    9251    9260    9272#
QDRV1   042050          9241    9252    9261    9273#
QDRV2   042070          9242    9253    9262    9274#
QDRV3   042110          9243    9254    9263    9275#
QDRV4   042130          9244    9255    9264    9276#
QDRV5   042150          9245    9256    9265    9277#
QDRV6   042170          9246    9257    9266    9278#
QDRV7   042210          9247    9258    9267    9279#
QINPT   041746          9240#   9314    9331*   9332*   9333    9335*
QOUTPT  041766          9251#   9314*   9352    9365    9366*   9367*   9368    9370*
QSTART  042006          9260#   9294    9299    9335    9370
QSTOP   042010          9261#   9333    9368
QTERM = 042230          9268    9280#
QUES    053566          6638    9573#
RANCYL  016506          5097#
RANPAT  016742          5117    5149#
```

```
RANSEC  016350          5052    5064#
RANSIZ  016622          5111    5122#   5130
RANTRK  016424          5080#
RANWRT  017066          5045    5190#
RANXIT  016752          5114    5146    5151#
RATIO   001500          2695#   3486*   5043    9583
RDCHR = 104410          7114    7151    7975#
RDDAT = 000171          2465#   2820    6513
RDHD  = 000173          2466#   2821    4795    5331
RDLIN = 104411          3408    6152    6479    6608    7976#   9684    9696    9767    9776
RDY   = 000200          2283#
RD.ADR  041230          9090*   9091*   9093#   9553
RD.RM   041204          8301    8320    8326    8548    8572    8586    8690    8704    8750    8794    8859    8869    8884
                        9021    9088#   9142    9156    9162    9195
RD.RM1  041226          9092#   9121
RD.RM2  041346          9088*   9120#
RD.RM3  041352          9099    9112    9122#
RD.RM4  041356          9104    9124#
RD.WRD  041232          9094#   9095    9553    9556
READDR  022262          3680    4026    4066    4101    5229    5296    5359    5573    5838#
READHD  015042          4027    4067    4102    4791#
READIN= 000121          2455#
RECAL = 000107          2450#   4760
RECALT  014764          4138    4759#
RECALO  014770          4760#   6242
REDAPK  025200          6186    6368#
REFMT   006136          3672#   3957    4019    4048    4093    4133    4169    4245
REFMTX  006256          3673    3675    3677    3679    3688    3692#
RELBUF  015554          3660    4921#
RELSE = 000113          2452#
REPLZ   027556          5965    5970    5976    5982    5988    5995    6000    6005    6010    6019    6093    6098    6103
                        6877#
RESREG= 104413          5009    6200    6812    7871    7921    7978#   8271    8404    8407    8470    8675    8735    9000
                        9204    9303
RESVEC= 000010          2260#
RETRY   001350          2630#   3886*   3907*   3924*   3925    3930*   3972*   3995*   4000*   4003*   4004    4040*   4085*
                        4125*   4161*   4189*   4258*   4288*   4305*   4819*   4831*   4832    5728
RMADR   034316          3346*   3352*   8179#   8251    8363    8487    8507    8529    8688    8729    9019    9090    9096
                        9100    9115    9148    9152    9184    9214    9781
RMAS  = 000016          8204#   8325*   8792    8908*   8935*   8956*   9041
RMBA  = 000004          8199#
RMCS1 = 000000          8197#   8293*   8299    8314    8427*   8428    8503    8525    8567    8587    8598    8613    8624
                        8751    8795    8846    8876    8913    8971    9143
RMCS2 = 000010          8201#   8252*   8292*   8294    8376*   8426*   8488*   8508*   8530*   8616    8635    8647    8660
                        8669*   8689*   8749*   8858*   8942*   8955*   9020*   9037*   9097    9153    9185*   9191    9215*
                        9220
RMDA  = 000006          8200#   8495    8521    8536
RMDB  = 000022          8206#   9189
RMDC  = 000034          8211#   8499    8511    8543
RMDS  = 000012          8202#   8321    8691    8860    8885    8943    9022
RMDT  = 000026          8208#   8302
RMEC1 = 000044          8215#
RMEC2 = 000046          8216#   9199
RMERRS  034150          8001#   8235    8829    8832    8842    8943*   8944*   8945*   8946*   9549
RMER1 = 000014          4068    4103    8203#   8327    8870    8944    9157
RMER2 = 000042          8214#   8945
```

```
RMHR   = 000036          8212#
RMINIT   034334          3422    8231#
RMLA   = 000020          8205#   8705    8707
RMMR1  = 000024          8207#
RMMR2  = 000040          8213#   8946
RMOF   = 000032          8210#   8318    8549    8553    8573    8577
RMR    = 000004          2354#
RMSN   = 000030          8209#
RMVEC    034320          3347*   3353*   8180#   8248    8250    8357
RMWC   = 000002          8198#
RM03     035040          4741    4761    4777    4797    5020    6525    8356#
RM03A    053155          3454    6431    9573#
RNOP   = 000101          2448#
RPTMR    040576          6126    8985#
RTC    = 000117          2454#   4740
RTNCTR   014740          4739#
S        051162          5578    5587    9573#
SAVEFG   034256          3423*   8130#   8602    8755    8932
SAVER1   001400          2647#   4420*   4470*   4515    4525    4529*
SAVERS   001402          2648#   4421*   4471*   4526    4530*
SAVREG=  104412          4984    6146    6802    7845    7906    7977#   8231    8359    8419    8634    8728    8988    919'
                         9287
SC       037442          8734    8763    8766    8772    8792#
SCMND    024716          6176    6306#
SC04   = 000400          2406#
SC1    = 000100          2404#
SC10   = 001000          2407#
SC11     040260          8831    8866    8923#
SC12     040370          8825    8836    8942#
SC13     040440          8815    8818    8952#
SC2    = 000200          2405#
SC20   = 002000          2408#
SC3      037506          8806#   8810
SC4      037512          8808#   8839    8850    8853    8901    8915    8922    8941    8978
SC5      037524          8807    8813#
SC6      037714          8835    8854#
SC6A     040024          8833    8878#
SC7      040152          8868    8881    8883    8893    8903#
SC8      040230          8847    8861    8871    8877    8886    8914    8916#   8951
SDETAL   022752          5911    5931    5954#
SEARCH=  000131          2457#
SECLMT   001442          2665#   3509    6450
SECOND   001370          2639#   3367*   6101    6112*   6113    6115*
SEEK   = 000105          2449#
SEEKFG   034260          8138#   8236    8451
SELDRV=  000145          2460#
SELPAR   016210          3615    5038#   5118
SETFMT=  000143          2459#
SETVEC   004650          3403    3405    3411    3415    3421#
SET.IE   041664          8265    8296    8394    8468    8674    8800    9213#
SHDTYP   022730          5904    5928    5941#
SIXTEE   001372          2640#   3364*   6109*   6111*
SIZMEM   004406          3375#
SKI    = 040000          2434#
SKIER    011752          3870    4270#
SLASH    054474          6430    6607    9592#
```

```
SPOTCK  020060          3894    3979    4024    4071    4111    4152    4174    5353#
SRCHWT  034232          8084#   8421*   8527*   8927    8936*
STACK = 001100          2164#   3285    3362
START   003622          2479    3272#   5891
START1  003632          2478    3274#
START2  003650          3273    3278#   7485
STATHO  053162          5945    9573#
STATIN  001314          2620#   3357*   3664    3666*   6131*
STATIS  015262          3648    4848#
STATPR  022614          3667    5900#   6323
STKLMT= 177774          2175#
STNDAT  002476          2913#   4595    4601    4999    5007
STO     040666          8995    9015#
STO1    040722          9026#
STO2    041012          9025    9027    9029    9041#
STO3    041042          9043    9049#
STO5    041066          9023    9055#
STO6    041074          9045    9057#
STO7    041132          9047    9065#
STO8    041162          9064    9071#
STO9    041172          9033    9040    9048    9052    9054    9056    9062    9069    9073#
SVRH70  041546          8604    8621    8652    8757    8769    8824    8873    8891    8934    8968    9036    9072    918:#
SWR     001154          2543#   3283    3299*   3301    3307*   3314*   3332    3426    3662    3672    3674    3707    3772
                        4406    4453    4553    4699    5039    5234    5314    5393    5396    5461    5525    5558    5757
                        5783    6180    6681    6716    6933    7123    7230    7236    7243    7255    7460    7473*   7650
                        7687*
SWREG   000176          2476#   3307    3332    7123    7650    7663
SWTIM   006620          3727    3779#
SW0   = 000001          2228#   3672    5039    6180
SW00  = 000001          2218#   2228
SW01  = 000002          2217#   2227    3707    4406
SW02  = 000004          2216#   2226    3426    3662
SW03  = 000010          2215#   2225
SW04  = 000020          2214#   2224    5314    6681    6716
SW05  = 000040          2213#   2223    5461    5525
SW06  = 000100          2212#   2222    5234
SW07  = 000200          2211#   2221    4553
SW08  = 000400          2210#   2220
SW09  = 001000          2209#   2219
SW1   = 000002          2227#
SW10  = 002000          2208#   5393
SW11  = 004000          2207#
SW12  = 010000          2206#
SW13  = 020000          2205#   5396    7230
SW14  = 040000          2204#
SW15  = 100000          2203#   3772    5757    5783
SW2   = 000004          2226#
SW3   = 000010          2225#   4453    4699
SW4   = 000020          2224#
SW5   = 000040          2223#   5558
SW6   = 000100          2222#
SW7   = 000200          2221#   3674
SW8   = 000400          2220#
SW9   = 001000          2219#
SYSTAT  053135          3430    9573#
T       051136          5575    5583    9573#
```

```
TABLE   054710          6453    9597#
TABLE0  054730          9597    9608#
TABLE1  054776          9598    9616#
TABLE2  055044          9599    9624#
TABLE3  055112          9600    9632#
TABLE4  055160          9601    9640#
TABLE5  055226          9602    9648#
TABLE6  055274          9603    9656#
TABLE7  055342          9604    9664#
TAB.XY= 001114          2515#   2522
TAP   = 040000          2399#
TBITVE= 000014          2261#
TD      037166          8732    8741#
THEAD   016270          5046    5051#
TIMER   034262          8143#   8340*   8463*   8607*   8653*   8744*   8855*   8899*   8938*   8953*   8974*   8991    8993*
                        9053*   9059*   9065*
TITLE   057150          3320    9837#
TKVEC = 000060          2268#   7104*   7105*
TPVEC = 000064          2269#
TRAPVE= 000034          2267#   3289*   3290*   3317*
TRE   = 040000          2288#
TRFER   011652          3829    4250#
TRKLMT  001444          2666#   3508    6448
TRNSWT  034230          8076#   8485*   8631*   8641    8658*   8667*   8746    8747*   9030
TRTVEC= 000014          2262#
TST1    003640          3276#
TYPDS = 104405          6606    6714    7970#
TYPE  = 104401          3320    3325    3406    3407    3429    3430    3437    3444    3446    3448    3450    3452    3453
                        3457    3459    3463    3480    3520    3562    3582    3588    3736    3737    3738    3744    3756
                        3757    3762    3763    3764    3770    3790    3791    3792    3798    5395    5886    5912    5962
                        5967    5972    5978    5984    5990    5997    6002    6007    6012    6090    6095    6100    6145
                        6151    6199    6301    6326    6327    6330    6331    6332    6341    6344    6346    6348    6354
                        6423    6430    6436    6438    6472    6478    6565    6566    6571    6575    6576    6581    6601
                        6607    6621    6638    6639    6645    6647    6660    6661    6662    6663    6670    6711    6712
                        6720    6721    6727    6729    6735    6750    6892    7118    7119    7127    7128    7135    7158
                        7164    7169    7173    7178    7183    7184    7186    7189    7193    7238    7246    7270    7287
                        7289    7292    7294    7298    7305    7353    7482    7554    7630    7661    7662    7665    7678
                        7689    7708    7966#   9683    9695    9761    9763    9766    9772    9775
TYPEST  022702          3563    5926#   6315    6736
TYPOC = 104402          7278    7302    7664    7967#   9765    9774
TYPON = 104404          7969#
TYPOS = 104403          3434    3585    3741    3767    3795    5959    6338    6427    6475    6568    6578    6642    6667
                        6724    6732    7968#
TYPRI4  027706          5942    5944    6920#
UCPAR   006536          3723    3756#
ULDFLG  034236          8103#   8291*   8366    8381*   8564*   8627*   8659*   8821    8905    8907*   8923    8925*   9039*
UNIT    001344          2628#   3696*   3739    3765    3793    5456    6715*   6722    6730
UNS   = 040000          2366#
UNSAF   012106          3823    4300#
UNTASN  053026          6211    6227    9573#
UNTMSG  052755          3582    6566    6576    6639    9573#
UNTNOT  053004          6297    6317    9573#
UNTOFF  052763          3448    6270    9573#
UNTON   052774          3452    9573#
UPE   = 020000          2312#
US1   = 000001          2299#
```

```
US2    = 000002          2300#
US4    = 000004          2301#
VV     = 000100          2339#
WAIT     001634          2765#   3554
WATPAK   025162          6197    6363#
WCE    = 040000          2313#
WCF    = 000040          2357#
WCFER    012010          3847    4291#
WCKD   = 000151          2461#   2816
WCKER    007766          3826    3962#
WCKHD  = 000153          2462#   2817
WCSEL    001476          2691#   5123    9582
WC.HK    037342          8758    8775#
WLE    = 004000          2363#
WLEER    011464          3853    4210#
WRL    = 004000          2344#
WRTDAT= 000161           2463#   2818    5335
WRTHD  = 000163          2464#   2819
WRTPK    017474          3513    3613    5287#
WRTPK1   017524          5289    5291    5295#
WRTPK2   017756          5294    5329#
WRTPK3   020002          5293    5328    5333#
WRTPK4   020034          5332    5338#
WRTPK5   020042          5312    5340#
WRT.AD   041450          9138*   9148*   9151#   9556
WRT.RM   041360          8313    8317    8494    8498    8502    8510    8520    8524    8535    8542    8552    8566    8576
                         8597    8612    8623    8845    8875    8912    9135#
WRT.R1   041444          9149#   9169
WRT.R2   041532          9135*   9168#
WRT.R3   041536          9144    9155    9158    9164    9170#
WRT.R4   041542          9160    9172#
WRT.R5   041544          9171    9173#
WRT.WD   041446          9136*   9140    9147*   9150#   9556
XWCNT    016562          5056    5060    5110#
ZROIND   001374          2641#   4419*   4463*   4469*   4473
$APTHD   001100          2502    2508#
$ASTAT= ****** U         7424    7439
$ATYC    031634          7395    7397#
$ATY1    031610          7393#
$ATY3    031616          7338    7394#
$ATY4    031626          7251    7396#
$AUTOB   001150          2540#   3336*   3349    3402    3484    7658    7748
$BASE    001260          2602#   3343    3345
$BDADR   001136          2535#
$BDDAT   001142          2537#
$BDSEC= 000124           5167    5356    6508    6537    9437#   9441
$BELL    001174          2553#   5395    7238    7261
$BUF   = 000006          3619*   4388    4410    4851    4930    4936    4941    4945    4987    5474    5623    5630    9387#
$CDW1    001264          2604#   6518*   6534*   6559*   6563    6573    6582
$CDW2    001266          2605#   6523*   6532    6560*
$CHARC   031604          7355*   7365*   7372    7381*   7386#
$CKSWR   032712          7650#   7974
$CMTAG   001114          2523#   3280    3281    3295
$CM3   = 000000          2553#
$CNTLC   031000          7119    7178    7203#
$CNTLG   033321          7128    7661    7743#
```

```
$CNTLU  033314          7173    7678    7742#
$CODE = 000024          4353    4402    4429    4479    4619    4713    4856    4858    4985    4989    5193    5236*   5244*
                        5259    5310*   5330*   5334*   6398*   6407    8775    8777    8785*   9398#
$COMND= 000002          4740*   4760*   4776*   4795*   5238*   5239    5243*   5246    5331*   5335*   6400*   6513*   8786*
                        8787*   9384#
$CPUOP  001232          2576#
$CRLF   001201          2555#   3406    3429    3459    3463    3736    3744    3756    3762    3770    3790    3798    4325
                        4332    4338    4344    4350    4545    4565    4577    4682    4694    4728    4730    5398    5418
                        5451    5453    5478    5494    5513    5533    5552    5567    5591    5614    5633    5654    5669
                        5682    5689    5722    5731    5755    5756    5781    5782    5912    5943    6090    6301    6348
                        6354    6438    6478    6571    6581    6647    6660    6661    6670    6711    6727    6735    6750
                        7118    7127    7183    7246    7261    7270    7289    7294    7298    7354    7389    7689    9761
$CYL  = 000012          4413    4793*   4864    4991    5026    5226    5254*   5301*   5302    5309*   5540    6461*   6514*
                        8783    9390#
$DBLK   032702          7596    7630    7638#
$DB2D   033546          5740    5748    5753    5768    5773    5969    5975    5981    5987    7077    7845#
$DB20   033742          7094    7906#
$DECVL  033726          7847    7893#
$DEVCT  001214          2567#   6758*
$DEVM   001262          2603#
$DOAGN  027270          6766    6771    6777#
$DRVID= 000224          6333    6485*   6486*   6487*   6488*   6489*   6490*   9441#   9445
$DSPLY  027734          6933#   7979
$DTBL   032672          7599    7634#
$ENDAD  027260          2488    6773#
$ENDCT  027244          6768#
$ENV    001224          2572#   3330    3338    3404    7248    7333    7402    7426
$ENVM   001225          2573#   3312    7335    7340    7404
$EOP    026722          6697#
$EOPCT  027236          6765#   6769
$ERFLG  001117          2526#   7233*   7261
$ERMAX  001131          2532#
$ERROR  031006          3287    7226#
$ERRPC  001132          2533#   7240*   7241*   7242    7261    7276
$ERRTB  003542          3218#   7284
$ERRTY  031172          7245    7269#
$ERTTL  001126          2530#   7239*   7261
$ESCAP= ****** U        8799    8849    8852    9106    9165    9374
$ETABL  001224          2571#
$ETEND  001270          2514    2606#
$FAIR = 000072          3622*   9413#
$FATAL  001206          2564#   7430*
$FFLG   032054          7393*   7396*   7424    7433*   7441#
$FILLC  001172          2551#   7358    7389
$FILLS  001171          2550#   7389
$FIRST= 000122          6444*   9433#
$FMT  = 000001          3928*   9383#   9384    9385    9386    9387    9388    9389    9390    9391    9392    9396
$GDADR  001134          2534#
$GDDAT  001140          2536#
$GET42  027250          3546    5889    6676    6770#
$GTSWR  032762          7132    7662#   7372
$HARD = 000062          6015    6044    6046*   9409#
$HD   = 000000          2139
$HIBTS  001100          2509#
$HINUM  033446          3293*   6790    7765    7773    7778*   7783#
$ICNT   001120          2527#
```

```
$ILLUP  032222          7452    7468    7487#
$INTAG  001151          2541#   7690    7748
$ITEMB  001130          2531#   7242*   7250    7261    7273
$LF     001202          2556#   7193    7261    7389
$LFLG   032053          7434*   7440#
$LKCSB  001276          2613#   5873*
$LKCSR  001274          2612#   5867    5874*
$LKS    001302          2615#   5878    5883*
$LLVEC  001304          2616#   5880
$LONUM  033450          3294*   5047    6789    7764    7771    7777*   7784#
$LPADR  001122          2528#
$LPERR  001124          2529#
$LPVEC  001300          2614#   5870
$LSTAD  055674          3378    9741*   9745#
$MADR1  001236          2589#
$MADR2  001242          2593#
$MADR3  001246          2596#
$MADR4  001252          2599#
$MAIL   001204          2510    2514    2552#   3277    3311    3330    7248    7333
$MAMS1  001234          2583#
$MAMS2  001240          2591#
$MAMS3  001244          2594#
$MAMS4  001250          2597#
$MBADR  001102          2510#
$MFLG   032052          7394*   7400    7435*   7439#
$MISPO= 000066          5776    6017    6068    6070*   9411#
$MNEW   033337          7665    7746#
$MSGAD  001220          2569#   7410*   7413
$MSGLG  001222          2570#   7415*
$MSGTY  001204          2563#   7408    7416*   7428    7432*
$MSWR   033326          7662    7744#
$MTYP1  001235          2584#
$MTYP2  001241          2592#
$MTYP3  001245          2595#
$MTYP4  001251          2598#
$NCODE= 000074          3685*   5050*   5110    5113    5138    5145    5196*   5224    5236    5237    9417#   9418    9419
                        9420    9421    9422    9423    9427
$NCYL = 000100          3681*   5054*   5055    5057*   5109*   5170    5254    9421#
$NEXT = 000104          3609    3691*   5151*   5262*   5338*   6389    9423#
$NPATC= 000075          3690*   5150*   5252    9418#
$NSEC = 000076          3683*   5053*   5058*   5076*   5253    9419#
$NTRK = 000077          3682*   5059*   5092*   9420#
$NULL   001170          2549#   7360    7389
$NWRDL= 000102          3684*   5112*   5144*   5255    5256    9422#
$NWTST= 000000          3275#
$OCNT   032462          7526*   7555*   7568#
$OCTVL  034044          7908    7933#
$OMODE  032464          7521*   7525*   7530    7533*   7544*   7570#
$OPERC= 000036          5024*   5025*   5288    5290    5738    5766    5956    6740    9403#
$PACK = 000026          3506*   3611    5292    5311    5327    5341*   6258*   9399#
$PASS   001212          2566#   3311*   6759*   6760*   6762*   6763*   6780
$PASSC= 000070          5963    6255*   6713    6718    6748*   9412#
$PASTM  001106          2512#
$PATTC= 000030          4998    5252*   5337*   6401*   6402*   9401#
$POSIT= 000042          4866*   4867*   5028*   5029*   5771    6706    6709    9404#
$POWER  032230          7483    7490#
```

```
$PREVA= 000032          5026    5226*   5227*   5230*   5231*   5232*   5297*   5298*   5299*   5581    5585    5589    5597
                        8783*   8784*   9402*
$PREVO= 000027          5222*   5223*   5241    5248    5295*   5428    8782*   9400*
$PSEL = 000003          9385*
$PWRAD  032216          7485*
$PWRDN  032056          3291    7452*   7480
$PWRMG  032212          7483*
$PWRUP  032130          7462    7468*
$QUES   001200          2554*   7186    7261    7389    7708
$RAND   033350          5038    5129    5209    5287    7760*
$RDCHR  033174          7721*   7975
$RDDEC= ****** U        7977
$RDLIN  030510          7146*   7976
$RDOCT= ****** U        7977
$RDSZ = 000001          7742*
$READ = 000052          4862*   4863*   5751    6700    6703    9406*
$REG  = 000014          6389    9391*   9471    9472    9479    9480    9487    9488    9495    9496    9503    9504    9511
                        9512    9519    9520    9527    9528
$RESRE  033510          7819*   7978
$RETRY  015134          3887    3973    4041    4086    4126    4162    4190    4260    4290    4306    4815*   4833
$RMADR  001270          2610*   3345*   3346    3352    9559    9762    9780
$RMAS = 000252          9452*   9565
$RMBA = 000240          4389    4609    4640    4661    4701    4721    4850    5475    5528    5629    5640*   5641    9447*
                        9565
$RMCS1= 000234          3699    3701    4365    5222    5295    5422    8782    9445*   9446    9447    9448    9449    9450
                        9451    9452    9453    9454    9455    9456    9457    9458    9459    9460    9461    9462    9463
                        9464    9471    9479    9487    9495    9503    9511    9519    9527    9561
$RMCS2= 000244          3824    3983    4034    4078    4118    4224    4239    4256    4367    9449*   9561
$RMDA = 000242          5053    5300    5561    5612    5843    5844    9448*   9565
$RMDB = 000256          5644    9454*   9565
$RMDC = 000270          3678    4106    4864    5054    5301    5565    5600    5845    9459*   9569
$RMDS = 000246          3703    3827    3856    9450*   9561
$RMDT = 000262          6443*   9456*   9565
$RMEC1= 000300          3875    3877    4625    5676    9463*   9561
$RMEC2= 000302          3879    4643    5680    6351    6394    9464*   9472    9480    9488    9496    9504    9512    9520
                        9528    9561
$RMER1= 000250          3830    3833    3836    3839    3842    3845    3848    3851    3854    3859    3862    3903    3920
                        3922    3962    3993    4006    4008    4063    4098    4369    4826    4829    9451*   9561
$RMER2= 000276          3865    3867    4371    9462*   9561
$RMHR = 000272          9460*
$RMLA = 000254          9453*   9565
$RMMR1= 000260          9455*   9565
$RMMR2= 000274          9461*   9561    9569
$RMOF = 000266          9458*   9569
$RMSN = 000264          9457*   9569
$RMVEC  001272          2611*   3342*   3347    3353    9771
$RMWC = 000236          4384    4412    4610    4703    5472    5531    5646    9446*   9565
$RTNAD  027272          6779*
$R2A  = ****** U        7979
$SAVRE  033452          7803*   7977
$SAVR6  032226          7461*   7469    7470*   7471*   7489*
$SB2D   030246          4636    5818    5964    5994    5999    6004    6009    6018    6092    6097    6102    7075*
$SB2O   030276          5802    7092*
$SEC  = 000010          4416    4792*   4994    5174    5227    5253*   5300*   5308*   5550    6459*   6524*   6531*   6532
                        6561*   8784    9380*
$SETUP= 000156          3263*   3286    3287    3289    3291    3293    3295    3327    6762    7227    7258    7260    7645
```

```
                        7748
$SIZE   055600          3377    9724#
$SKI  = 000064          5779    6016    6056    6058*   9410#
$SOFT = 000060          6014    6032    6034*   9408#
$SSEC = 000022          4422    4427    4428    4613    4705    5258*   5261*   5329*   5333*   5649    6406*   6409*   9397#
$STUP = 177777          3263#
$SUPRS  027646          4637    5741    5749    5754    5769    5774    5819    6904#
$SVPC = 000210          2486#   2491
$SWR  = 122000          2128#   2139    2143    2144    2145    2146    2553    3277    3295    6694    6762    6772    6778
                        6780    7219    7220    7221    7222    7223    7236    7243    7255    7259    7261    7486
$SWREG  001226          2574#   3314
$STATUS= 000016         3644    3697    3714    3720    3722    3724    3726    3728    3730    3821    3912    3915    3932
                        3997    4764    4800    4816    4822    4848    5469    5620    6528    9392#   9569
$TERM = 000032          7980#
$TESTN  001210          2565#   3277*   6780*
$TIME   023450          5400    5941    6088#   6149    7232
$TKB    001162          2546#   6150    6201    7106    7643    7654    7671    7725    7731
$TKINT  030326          3326    3482    7104#
$TKS    001160          2545#   6202*   7107*   7121*   7131*   7133*   7643    7652    7668    7692*   7723    7729
$TKSRV  030356          7104    7114#
$TN   = 000002          2128#   2139    3275    3277#
$TNPWR  033656          7852    7853    7873#
$TOTAL= 000056          5743    6013    6080    6082*   6683    9407#
$TPB    001166          2548#   7378*   7389
$TPFLG  001173          2552#   7327    7389
$TPS    001164          2547#   7376    7389
$TRANS= 000046          4854*   4855*   4860*   4861*   5746    9405#
$TRAP   034062          3289    7943#
$TRAP2  034104          7954#   7965
$TRK  = 000011          4791*   5172    5305    5307*   5545    6460*   6515*   9389#
$TRP  = 000015          7958#   7967#   7968#   7969#   7970#   7971#   7972    7973#   7974    7975#   7976#   7977#   7978#
                        7979#   7980#
$TRPAD  034116          7948    7965#   7980
$TSTM   001104          2511#
$TSTNM  001116          2525#   7235    7261
$TTYIN  030766          7148    7149    7161    7184    7198    7202#
$TYPBN= ****** U        7971
$TYPDS  032466          7584#   7970
$TYPE   031326          6923    6936    7327#   7421    7958    7966
$TYPEC  031540          7357    7364    7371    7376#   7377    7694
$TYPEX  031606          7382    7384    7387#
$TYPOC  032264          7524#   7967
$TYPON  032300          7523    7526#   7969
$TYPOS  032240          7519#   7968
$UNIT   001216          2568#   2626
$UNITM  001110          2513#
$USWR   001230          2575#
$VECT1  001254          2600#   3340    3342
$VECT2  001256          2601#
$WRDL = 000020          4386    4411    4611    4702    4860    4886    4893    4895*   4896    4897*   4928    4937    4942
                        4946    4988    5255*   5318*   5319*   5325*   5471    5626    5647    6403*   9396#   9397    9398
                        9399    9400    9401    9402    9403    9404    9405    9406    9407    9408    9409    9410    9411
                        9412    9413    9417
$WROM = 000004          4796*   5256*   5257*   5320*   5321*   5322    5324*   5326*   6404*   6405*   6516*   9386#
$SGET4= 000000          6772#
$OFILL  032463          7520*   7524*   7534    7569#
```

```
$40CAT= ****** U        7245
.     = 057527          2470#   2474#   2486    2487#   2489#   2491#   2492#   2498    2499#   2501#   2503#   2515    2522#
                        2557    2841#   2894#   2900#   3284    3341    3344    3606    3626    3822    3825    3831    3834
                        3837    3840    3843    3846    3849    3852    3860    3863    4641    6780    6852    7202#   7205#
                        7231    7261    7308#   7389    7442#   7464    7488    7638#   7643    7748    7893#   7933#   7980
                        9272#   9273#   9274#   9275#   9276#   9277#   9278#   9279#   9280    9470#   9471    9472#   9478#
                        9479    9480#   9486#   9487    9488#   9494#   9495    9496#   9502#   9503    9504#   9510#   9511
                        9512#   9518#   9519    9520#   9526#   9527    9528#   9537#   9545#   9573#   9714#   9832#   9833
.$ASTA= ****** U        7394    7397
.$X   = 001100          2498#   2503
```

A	6024#	6025	6037	6049	6061	6073									
CKCHR	2128#	7026	7040												
CKDIG	2128#	6612													
CKNUM	2128#														
COMMEN	2271#														
ENDCOM	2271#														
ERENTR	7213#	7228													
ERRCAL	7984#	8799	8849	8851	9105	9165									
ERROR	2165#	8799	8849	8852	9106	9165									
ESCAPE	2271#														
GETPRI	2271#	9730													
GETSWR	2128#	2271#	3327												
MORETA	2492#	2608													
MULT	2271#														
NEWTST	2271#	3275													
POP	2271#	4531	5182	5379	5481	6410	6585	7436	7437	7473	7474	7625	7779	7824	
PUSH	2271#	4522	5161	5353	5414	6377	6502	7397	7399	7420	7454	7460	7584	7760	7804
REPORT	2271#														
RM03.D	1#	7984													
SCOPE	2166#														
SETPRI	2271#														
SETTRA	7958#	7967	7968	7969	7970	7972	7974	7975	7976	7977	7978	7979			
SETUP	2271#	3279													
SKIP	2271#														
SLASH	2271#														
SPACE	2271#														
STARS	2271#	2274	2278	2320	2324	2442	2446	2484	2495	2497	2504	2518	2557	2560	2669
	2673	2725	2729	2741	2745	2907	2911	3263	3270	3275	3524	3528	5384	5388	5823
	5827	6692	7209	7215	7264	7312	7392	7450	7466	7496	7574	7642	7645	7713	7751
	7788	7834	7897	7937	7982	7993	9375	9379	9539	9543	9674	9718			
SWRSU	2271#	3295#													
TRMTRP	7958#	7980													
TYPBIN	2271#														
TYPDEC	2271#														
TYPNAM	2271#														
TYPNUM	2271#														
TYPOCS	2271#	3431	3583	3739	3765	3793	5957	6336	6425	6640	6665	6722	6730		
TYPOCT	2271#	7276	7300	7663											
TYPTXT	2271#														
XXEP	6690#	6698													
$$CMRE	2516#														
$$CMTM	2516#														
$$ESCA	2271#														
$$NEWT	2271#	3275													
$$SET	7958#	7967	7968	7969	7970	7972	7974	7975	7976	7977	7978	7979			
$$SETM	3311#														
$$SKIP	2271#														
.EQUAT	2128#	2161													
.HEADE	2128#	2129													
.SETUP	2128#	3263													
.SWRHI	2128#	2139													
.SWRLO	2128#	2146#	2147	2148	2150	2152	2157	2159	2160						
.$ACTI	2128#	2482													
.$APTB	2128#	2558#													
.$APTH	2128#	2493													
.$APTY	2128#	7390													

```
.$CATC   2128#   2468
.$CMTA   2128#   2516
.$D82D   2128#   7832
.$D82O   2128#   7895
.$EOP    2128#   6690
.$ERRO   2128#   7213
.$ERRT   2128#   7262
.$POWE   2128#   7448
.$RANO   2128#   7749
.$READ   2128#   7640
.$SAVE   2128#   7786
.$SIZE   2128#   9716
.$TRAP   2128#   7935
.$TYPD   2128#   7572
.$TYPE   2128#   7310
.$TYPO   2128#   7494


. ABS.  057527     000


 ERRORS DETECTED:  0

 DSKZ:DZRMBA,DSKZ:DZRMBA.SEQ/NL:MD:MC:CND:TOC/DOC/SOL/CRF=DSKM:RMDRV4.P11,DSKM:DZRMBA.P11
 RUN-TIME: 35 27 2 SECONDS
 RUN-TIME RATIO: 282/65=4.3
 CORE USED:  42K  (83 PAGES)

 DOCUMENT PAGES:  209
```