RP04

MULTI-DRIVE EXERCISER

MD-11-DZRPN-B

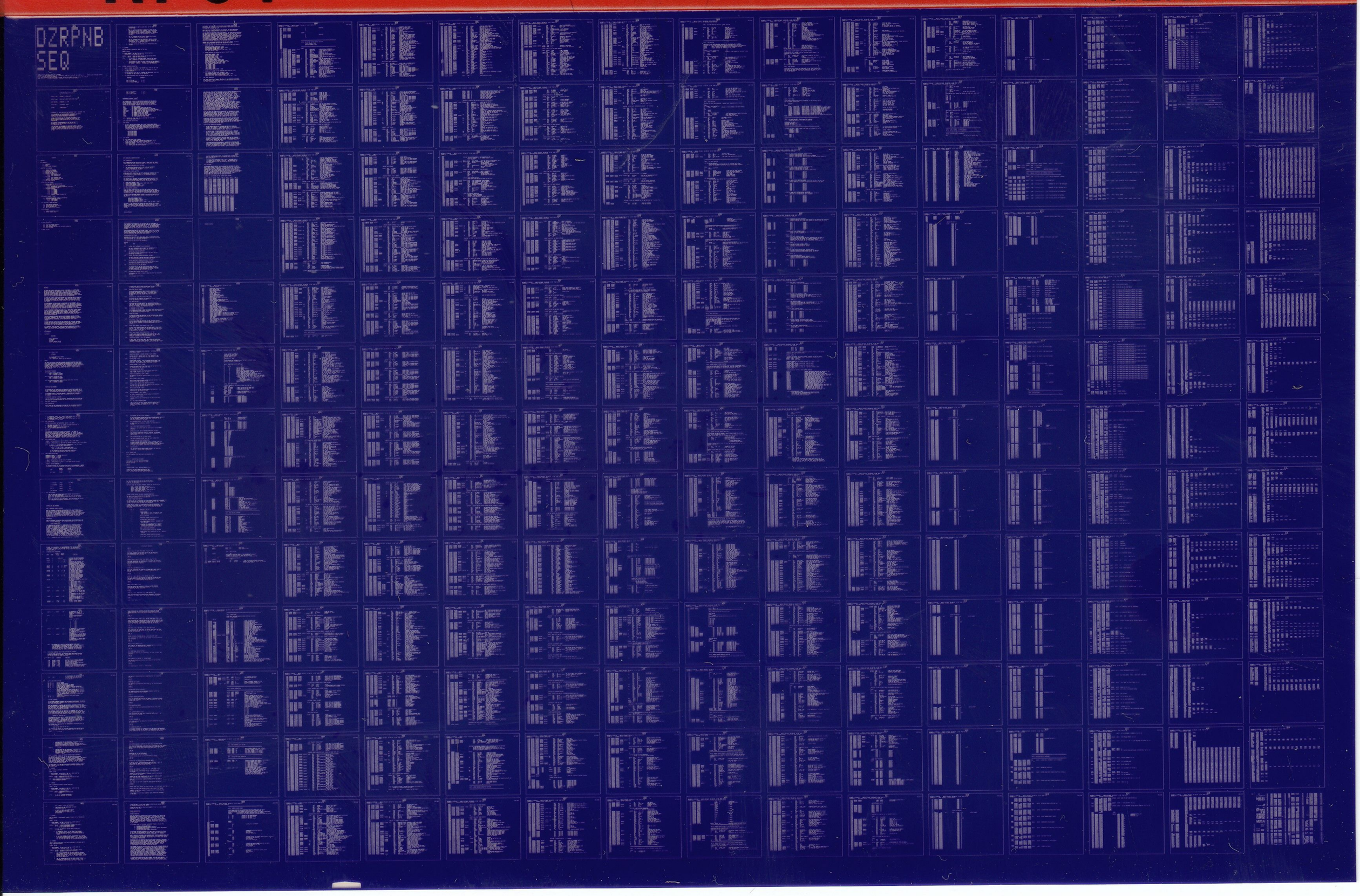
EP-DZRPN-B-DL-A

MAR 1976

digital

Made In U.S.A.

FICHE 1 OF 2



RP04

MULTI-DRIVE EXERCISER

MD-11-DZRPN-B

EP-DZRPN-B-DL-A

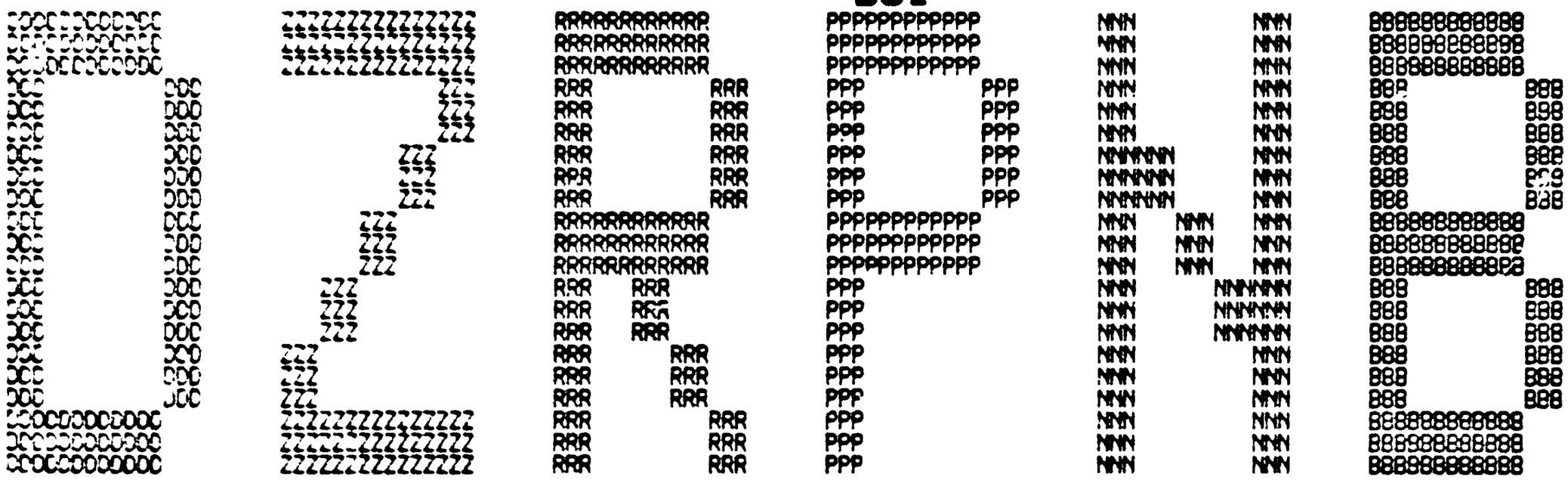
MAR 1976

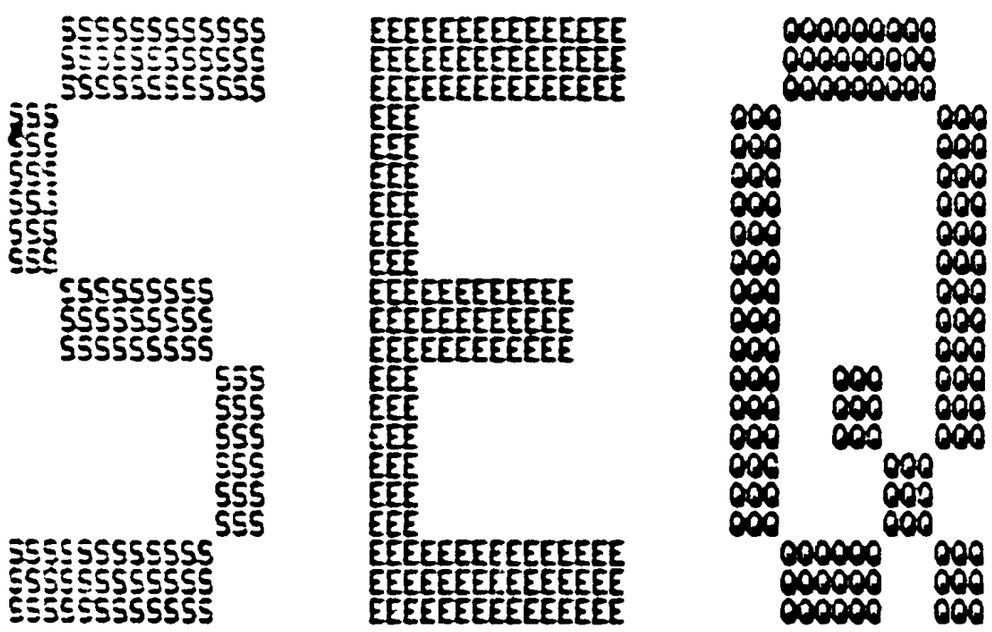
FICHE 2 OF 2

digital

Made In U.S.A.

B01





```
_PTSP_ version 101(2111) Running on MTA260
 *START* User WELLINGHAM,A[1,2]  Job  DZRPNB Seq. 1905 Date 13-Feb-76 18:15:28 Monitor     RZ231D KL10 SYS#1026 *START*
Request created: 13-Feb-76 18:08:07
File: WHTN:DZRPNB.SEQ<055>[404,3722] Created: 13-Feb-76 15:08:51 Printed: 13-Feb-76 18:15:29
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:486 /FORMS:NORMAL
```

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

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

PRODUCT NAME: RP04 MULTI-DRIVE EXERCISER PROGRAM

DATE CREATED: FEBRUARY 21, 1975

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: CHARLES HESS

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 STARTUP

4. CONTROLLING THE PROGRAM

    4.1 DATE & OPERATOR IDENTIFICATION
    4.2 PARAMETERS
        4.2.1 KEYBOARD ENTRY PARAMETERS
        4.2.2 PERIPHERIAL ADDRESS PARAMETER LOCATIONS
    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 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

E01

9. PROGRAM DESCRIPTION

SEQ 0003

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. PROGRAM LISTING

1. ABSTRACT

THE RP04 MULTIDRIVE EXERCISER PROGRAM EXERCISES 1 TO 8 RP04 DISK DRIVES ATTACHED TO THE SAME RH11. IF 2 OR MORE RP04 DISK DRIVES ARE BEING EXERCISED, OPERATIONS ON THE DRIVES ARE OVERLAPPED (OTHER DRIVES ARE PERFORMING SEEK/SEARCH OPERATIONS WHILE ONE RP04 IS PERFORMING A DATA TRANSFER OR WRITE CHECK OPERATION). OPERATIONS AMONG THE RP04'S ARE OPTIMIZED SO THAT A HIGH SUBSYSTEM DATA TRANSFER RATE OR A HIGH POSITIONING OPERATION RATE IS MAINTAINED.

THE RP04 MULTIDRIVE EXERCISER PROGRAM WILL EXERCISE RP04'S CONNECTED AS EITHER SINGLE OR DUAL PORT UNITS. DUAL PORT RP04'S ARE TESTED BY LOADING AND RUNNING THE PROGRAM FROM BOTH CONTROLLING SYSTEMS. THE PROGRAM WILL EXERCISE A MIXED SYSTEM OF DUAL PORT AND NON DUAL PORT RP04'S.

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 PROCESSING.

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.

PROGRAM/OPERATOR COMMUNICATIONS ARE THROUGH THE TELETYPE; PROGRAM OPTIONS ARE SELECTED BY SWITCH REGISTER SETTINGS. ERROR ARE NORMALLY REPORTED ON THE TELETYPE; HOWEVER, IF A LINE PRINTER IS AVAILABLE THE PROGRAM WILL USE THE PRINTER FOR ERROR MESSAGE DISPLAY.

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
TELETYPE
PROGRAM LOADING DEVICE
KW11-L OR KW11-P CLOCK

RH11 WITH 1 RP04

OPTIONAL

4K TO 12K ADDITIONAL MEMORY
LINE PRINTER
1 TO 7 ADDITIONAL RP04'S ON THE SAME RH11

2.2 MEDIA

THE RP04 MULTIDRIVE EXERCISER PROGRAM REQUIRES FORMATTED DISK PACKS WHICH CONTAIN RANDOM OR PATTERNED DATA RECOGNIZED BY THE EXERCISER. DISK PACKS USED BY THE PROGRAM MAY BE GENERATED BY THE RP04 FORMATTER PROGRAM (MAINDEC-11-DZRPL) OR BY THE 'W' COMMAND OF THE RP04 MULTIDRIVE EXERCISER (SEE SECTION 4.4). THE PACKS MUST BE FORMATTED IN 22 SECTOR (16 BIT) MODE; THE ALTERNATE (20 SECTOR - 18 BIT) MODE IS NOT SUPPORTED.

2.3 PRELIMINARY PROGRAMS

RP04 DISKLESS CONTROLLER TEST
PART 1 (MAINDEC-11-DZRPS)
PART 2 (MAINDEC-11-DZRPT)

RP04 FUNCTIONAL CONTROLLER TEST
PART 1 (MAINDEC-11-DZRPU)
PART 2 (MAINDEC-11-DZRPV)

RP04 DUAL CONTROLLER LOGIC TEST (FOR DUAL PORT DRIVE TESTING)
PART 1 (MAINDEC-11-DZRPP)
PART 2 (MAINDEC-11-DZRPQ)

3. OPERATING THE PROGRAM

3.1 THE PROGRAM MAY BE LOADED WITH THE ABSOLUTE PAPER TAPE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE MEDIA USING THE ASSOCIATED 'XXDP' LOADER. THE PROGRAM CAN BE INSTRUCTED TO PRESERVE EITHER LOADER TYPE.

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.

3.3 THE RESTART LOCATION IS 204(8)

3.4 ONCE THE PROGRAM IS LOADED AND STARTED, OPERATIONS ARE INITIATED AND CONTROLLED BY KEYBOARD COMMANDS AND SWITCH REGISTER SWITCH SETTINGS.

3.5 PASS/TEST TERMINATION

3.5.1 PASS TERMINATION

END OF PASS MAY BE DETERMINED BY EITHER OF THE FOLLOWING CONDITIONS. THE END OF PASS CONDITION USED IS DETERMINED BY PARAMETER 'ENDET'.

A. IF PARAMETER 'ENDET' 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 'ENDET' 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 'PASCNT'.
B. THE TOTAL ERRORS ACCUMULATED EXCEED 100.
C. A FATAL ERROR OCCURS: EM12 OR EM14.

3.6 RUN TIME

THE EXERCISER PROGRAM MAY BE RUN IN TWO MODES. THE MODE IS DETERMINED BY THE VALUE IN PARAMETER 'MAXDL'. IF 'MAXDL' IS ONE SECTOR, THE PROGRAM RUNS IN A SEEK HEAVY MODE; IF 'MAXDL' APPROACHES ONE TRACK IN SIZE (5720 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 2.5 HRS (TO REACH 1.875 X 10↑8 WORDS)
TO
8 DRIVES - APPROXIMATELY 11 HRS (FOR ALL DRIVES TO REACH 1.875 X 10↑8 WORDS)

NOTE: IF SW<01> = 1 (NO SOFTWARE DATA COMPARSIONS), THE TUN TIMES ARE THE FOLLOWING VALUES, APPROXIMATELY:

1 DRIVE - 1.7 HRS (1.875 X 10↑8 WORDS READ)
ADD 1/2 HOUR FOR EACH ADDITIONAL DRIVE TESTED.

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 'MAXDL' = 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/RP04	176700	254

TTY PRINTER	177564	NOT USED
TTY KEYBOARD	177560	60
KW11-L	177546	100
KW11-P	172542	104
LINE PRINTER	177514	NOT USED

3.8 DUAL PORT STARTUP

A. LOAD THE RP04 MULTIDRIVE EXERCISER PROGRAM (REVISION B OR LATER) INTO BOTH PROCESSORS.
B. SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'A/B' ON EACH RP04 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 THROUGH EACH PROCESSOR WERE RUNNING INDEPENDENTLY OF THE OTHER.

4. CONTROLLING THE PROGRAM

4.1 DATE & OPERATOR IDENTIFICATION

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, THE PROGRAM WILL ASK FOR A DATE ENTRY AND FOR AN OPERATOR I.D. ENTRY. 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 OPERATION IDENTIFICATION MAY BE ENTERED. BOTH THE DATE AND THE OPERATOR I.D. WILL BE TYPED WHEN THE 'SA' COMMAND IS PERFORMED (SEE SECTION 4.4.3).

4.2 PARAMETERS

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, 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 OR AN 'N' IF NO ENTRIES WILL BE MADE. THE PROGRAM WILL IDENTIFY THE PARAMETER BY THE NAME GIVEN BELOW, DISPLAY THE CURRENT VALUE OF THE PARAMETER AND THE BASE OF THE PARAMETER (OCTAL OR DECIMAL), AND WAIT FOR THE ENTRY. IF ONLY A CARRIAGE RETURN IS ENTERED, THE PROGRAM WILL USE THE PRESENT VALUE OF THE PARAMETER. THE RUBOUT KEY WILL ALLOW THE OPERATOR TO DELETE AN INCORRECT ENTRY; 'CONTROL U' WILL ALLOW AN ENTIRE ENTRY TO BE DELETED AND REENTERED. THE PROGRAM WILL TYPE A '?' IF AN ALPHABETIC OR INCORRECT NUMBER (FOR THE BASE OF THE PARAMETER) IS ENTERED. IF 'CONTROL C' IS ENTERED, THE PROGRAM WILL USE THE PRESENT VALUES OF THE REMAINING PARAMETERS AS DEFAULT VALUES.

NOTE: A PARAMETER ENTRY MUST BE TERMINATED BY A 'CARRIAGE

RETURN' TO BE ACCEPTED. IF THE PARAMETER ENTRY IS FOLLOWED BY A 'CONTROL C' INSTEAD OF A 'CARRIAGE RETURN', THE PROGRAM WILL RECOGNIZE ONLY THE 'CONTROL C' AND THE NEW PARAMETER VALUE WILL NOT BE ACCEPTED.)

4.2.1 KEYBOARD ENTRY PARAMETERS

NAME	BASE	DEFAULT VALUE	VALUE RANGE	FUNCTION
MAXDL	10	(SEE NOTE)		CONTROLS THE MAXIMUM BUFFER SIZE USED FOR DATA TRANSFERS
PASCNT	10	1	1 - 999	NUMBER OF PASSES TO END OF TEST.
INTRVL	10	5	0 - 256	DETERMINES THE INTERVAL (IN MINUTES) BETWEEN AUTOMATIC PERFORMANCE SUMMARY TYPEOUTS ERRORS PRINTED OUT IF SW<07>=0
CMPLMT	10	3	0 - 'MAXDL'	THE NUMBER OF DATA COMPARSION
MAXCYL	10	410	0 - 410	THE MAXIMUM CYLINDER ADDRESS
MINCYL	10	0	0 - 410	THE MINIMUM CYLINDER ADDRESS. VALUE MUST NOT BE GREATER THAN VALUE IN 'MAXCYL'
MAXTRK	10	18	0 - 18	THE MAXIMUM TRACK ADDRESS
MINTRK	10	0	0 - 18	THE MINIMUM TRACK ADDRESS THE MINIMUM TRACK ADDRESS MUST NOT BE GREATER THAN THE VALUE IN 'MAXTRK'
MAXSEC	10	21	0 - 21	THE MAXIMUM SECTOR ADDRESS
MINSEC	10	0	0 - 21	THE MINIMUM SECTOR ADDRESS THE MINIMUM SECTOR ADDRESS MUST NOT BE GREATER THAN THE VALUE IN 'MAXSEC'
BEGCOD	10	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
BEGPAT	10	8	1 - 15	THE CODE FOR THE STARTING PATTERN. (IF A WRITE ORDER OR A WRITE CHECK ORDER IS SPECIFIED IN 'BEGCOD')
ENDET	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
SRPADR	8	176700	N/A	THE RH11 UNIBUS ADDRESS
SRPVEC	8	254	N/A	THE RH11 VECTOR ADDRESS

SAVLOD	8	000001	0 OR 1	IF PARAMETER = 0, DO NOT PRESERVE THE LOADER; IF PARAMETER > 0, PRESERVE THE LOADER
RATIO	8	3	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

AUTOCK	8	000001	0 OR 1	IF PARAMETER = 1, THE PROGRAM PERFORM WRITE CHECKS AFTER EACH WRITE COMMAND. IF PARAMETER = 0, THE PROGRAM WILL PERFORM WRITE CHECKS RANDOMLY.
NOTPRT	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.

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 5980 (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
1166	$LKCSR	172540	ADDRESS OF KW11-P STATUS REGISTER
1170	$LKCSB	172542	ADDRESS OF KW11-P COUNTER BUFFER
1172	$LPVEC	104	KW11-P VECTOR ADDRESS
1174	$LKS	177546	ADDRESS OF KW11-L STATUS REGISTER
1176	$LLVEC	100	KW11-L VECTOR ADDRESS
1206	$LSCS	177514	ADDRESS OF LINE PRINTER STATUS REGISTER
1210	$LSDB	177516	ADDRESS OF LINE PRINTER DATA BUFFER
1214	HZ	74	74 (60 DECIMAL) IF SYSTEM IS 60 HZ;

1216	LA30	0	62 (50 DECIMAL) IF SYSTEM IS 50 HZ. 0 IF AN LA30 IS ON THE SYSTEM; 177777 IF MOD 33 OR MOD 35 TTY ON THE SYSTEM	SEQ 0010

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 <1> = 1 INHIBIT DATA COMPARSION AFTER READ ORDERS
SW <0> = 1 READ ONLY MODE

4.4 KEYBOARD COMMANDS

THE KEYBOARD COMMANDS CONTROL THE ASSIGNMENT/DEASSIGNMENT OF DRIVES, ALLOW THE OPERATOR TO REQUEST DRIVE PERFORMANCE SUMMARIES, AND ALLOW DATA PACKS TO BE WRITTEN

WHEN THE PROGRAM IS STARTED (OR RESTARTED), KEYBOARD COMMANDS ARE NOT RECOGNIZED UNTIL THE PROGRAM HAS INITIALIZED ITSELF. THE OPERATOR MUST WAIT UNTIL THE 'PROGRAM INITIALIZE COMPLETE' MESSAGE IS TYPED BEFORE ATTEMPTING TO USE THE COMMANDS. AFTER INITIALIZATION HAS BEEN COMPLETED, THESE COMMANDS MAY BE USED.

EACH COMMAND, EXCEPT THE 'S' AND THE 'D' COMMANDS, WILL ASK FOR A DRIVE IDENTIFICATION ENTRY AND FOR THE ADDRESSES OF ANY BAD SPOTS ON THE PACK BEING USED ON THE DRIVE WHICH IS BEING ASSIGNED. 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 'CARRIAGE RETURN' ONLY (NULL ENTRY) IN RESPONSE TO THE I.D. REQUEST.

THE PROGRAM WILL THEN ASK FOR THE ADDRESSES OF KNOWN BAD SPOTS ON THE DISK PACK USED:

'BAD TRK/SEC ADRS FOR DRV #N ?'

THE OPERATOR MAY SPECIFY UP TO 8 BAD LOCATIONS FOR THE PACK BEING USED. THE FORMAT FOR THE BAD SPOT ADDRESS ENTRY IS AS FOLLOWS:

FORMAT 1: C,T,S<CR>

A. LEADING ZEROS ARE NOT REQUIRED. THE ENTRY MUST BE TERMINATED BY A 'CARRIAGE RETURN' (<CR>).
B. THE PROGRAM WILL INHIBIT DATA ERROR MESSAGES OR WILL IDENTIFY DATA ERRORS WHICH OCCUR AT THE SPECIFIED ADDRESS, DEPENDING ON THE VALUE OF PARAMETER 'NOTPRT'.

FORMAT 2: C,T<CR>

A. LEADING ZEROS ARE NOT REQUIRED. THE ENTRY MUST BE TERMINATED BY A 'CARRIAGE RETURN' (<CR>).
B. WHEN THIS FORMAT IS USED, THE ENTIRE TRACK WILL BE CONSIDERED BAD; DATA ERRORS WILL BE HANDLED AS IN 'FORMAT 1', ABOVE.

NOTE: CYLINDER, TRACK, AND SECTOR ENTRIES ARE IN DECIMAL.

THE OPERATOR MAY BYPASS ALL OF THE BAD TRACK/SECTOR ADDRESS ENTRIES BY ENTERING A 'CONTROL C' INSTEAD OF AN ADDRESS. THE ADDRESS ENTRY MAY BE TERMINATED AT ANY POINT BY ENTERING A 'CONTROL C', ALSO. NOTE THAT AN ENTRY MUST BE TERMINATED BY A 'CARRIAGE RETURN' TO BE ACCEPTED BY THE SYSTEM. THE 'CONTROL C' MUST BE ENTERED IN PLACE OF AN ADDRESS ENTRY AND NOT IN PLACE OF THE 'CARRIAGE RETURN' TERMINATOR FOR AN ENTRY.

FOR BOTH DRIVE I.D. AND BAD TRACK/SECTOR ADDRESS ENTRIES, 'RUBOUT' AND 'CONTROL U' MAY BE USED TO CORRECT MIS-TYPED CHARACTERS AND MIS-TYPED LINES.

4.4.1 'T' COMMAND

USED TO ASSIGN A DRIVE(S) FOR TEST

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, ONLY DRIVES BEING TESTED WILL BE DEASSIGNED - THE ERROR MESSAGE IN (1) ABOVE WILL NOT BE DISPLAYED.

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 'INTRVL' IS NOT ZERO, THE PROGRAM WILL AUTOMATICALLY DISPLAY A PERFORMANCE SUMMARY FOR EACH DRIVE BEING TESTED AT A RATE DETERMINED BY 'INTRVL'.

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 RP04 MULTI-DRIVE 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 PACKS GENERATED BY THE RP04 FORMATTER PROGRAM (MD-11-DZRPL) OR BY THE RP04 MECHANICAL & READ/WRITE PROGRAM (MD-11-DZRPK), TEST 15, ARE ACCEPTABLE. (PACKS WRITTEN BY TESTS 13, 14 OR 16 OF 'DZRPK' CANNOT BE USED AND MUST BE REWRITTEN.)

2. THE 'W' COMMAND SHOULD NOT BE USED UNLESS 'MAXDL' PARAMETER IS APPROXIMATELY 1 TRACK IN SIZE - 5000 (10). IF THE BUFFER SIZE IS MUCH LESS THAN 1 TRACK, THE

TIME REQUIRED TO WRITE A DATA PACK IS TOO GREAT TO BE PRACTICAL.

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 SET SET DURING THE OPERATION OF THE 'W' COMMAND, THE DRIVE PERFORMING THE 'W' COMMAND WILL IGNORE THE SWITCH.

6. THE DATA WRITTEN IS VERIFIED BY A 'WRITE CHECK DATA' COMMAND.

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.
RD<CR> - READ THE PACK ON DRIVE 0.

NOTES: 1. THE PROGRAM WILL PERFORMA 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 GENERAL COMMAND INFORMATION

A. WHEN A COMMAND IS ENTERED, THE PROGRAM WILL TYPE OUT THE TIME

B. IF THE COMMAND ENTERED IS NOT VALID, THE PROGRAM WILL TYPE 'INVALID COMMAND'.

C. DRIVES ASSIGNED (WITH THE 'T' COMMAND) OR DEASSIGNED (WITH THE 'D' COMMAND) MAY BE ENTERED IN ANY SEQUENCE.

D. THE ERROR RESPONSES FROM THE PROGRAM ARE AS FOLLOWS

RESPONSE	COMMAND(S)
?UNIT N OFFLINE	T, W, R
?UNIT N NOT ASSIGNED	D, S
?UNIT N ALREADY ASSIGNED	T, W, R

?UNIT N NOT PRESENT	T, W, R	
?UNIT N UNSAFE	T, W, R	
?UNIT N NOT AN RP04	T, W, R	

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' OR 'OCYL' ERRORS
'MISP'	THE NUMBER OF POSITIONING ERRORS
'OTHER'	THE TOTAL ERRORS OF OTHER TYPES

NOTE: ERRORS EM1, EM2, EM3, EM4, & 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 CORRECTED OR BECOMES 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 AT EACH OF THE FOLLOWING OFFSETS:

+400 MICRO-INCHES
-400 MICRO-INCHES
+800 MICRO-INCHES
-800 MICRO-INCHES
+1200 MICRO-INCHES
-1200 MICRO-INCHES

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

AFTER EACH WRITE OPERATION, THE DATA WRITTEN IS CHECKED WITH THE APPROPRIATE WRITE CHECK COMMAND - IF PARAMETER 'AUTOCK' IS 1. (IF 'AUTOCK' IS 0, WRITE CHECKS WILL BE PERFORMED RANDOMLY.)

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 RP04 SUBSYSTEM DOES NOT HAVE AN AUTOMATIC BAD TRACK HANDLING CAPABILTIY, THE MULTIDRIVE EXERCISER ALLOWS THE OPERATOR TO IDENTIFY UP TO 8 BAD TRACK/SECTOR LOCATIONS WHEN THE DRIVE IS ASSIGNED FOR TEST. (SEE SECTION 4.1 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 IDENTIFY 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 OCCURED OR A CENTRAL PROCESSOR FAILURE HAS OCCURED.

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	ILLEGAL RH11 INTERRUPT (SC=0 OR RHAS=0) THE RH11 INTERRUPTED AND EITHER 'SC' FOR NOT SET OR RHAS WAS ZERO AND NO RH11 ERROR WAS INDICATED.
EM2	UNEXPECTED ATTENTION DETECTED THE INDICATED DRIVE INTERRUPTED BUT THE DRIVE WAS NOT PERFORMING AN OPERATION.
EM3	CONTROL BUS PARITY ERROR DETECTED BY THE RH11 THE RH11 DETECTED A CONTROL BUS PARITY ERROR WHEN READING THE INDICATED REGISTER FROM THE INDICATED DRIVE.
EM4	CONTROL BUS PARITY ERROR DETECTED BY THE RP04 THE INDICATED RP04 DETECTED A CONTROL BUS PARITY ERROR WHEN THE RH11 LOADED THE SPECIFIED REGISTER.
EM5	ATTENTION FROM AN OFFLINE DRIVE THE ATTENTION BIT WAS SET FOR A DRIVE WHICH WAS NOT AVAILABLE. THIS MESSAGE DOES NOT NECESSARILY INDICATE AN ERROR CONDITION. THIS MESSAGE WILL OCCUR IF DRIVES NOT BEING EXERCISED ARE CYCYLED UP OR DOWN.
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 OCCURED 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.

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 OCCURED 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 OCCURED 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 OCCURED. 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 OCCURED. 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: 'IXE', 'AOE', 'RMR', 'ILF', OR 'ILR'

EM31 OPERATION INCOMPLETE ('OPI') ERROR

AN OPERATION INCOMPLETE ERROR OCCURED AT THE INDICATED SECTOR.

EM32 DRIVE TIMING ('DTE') ERROR

DRIVE TIMING ERROR OCCURED 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 OCCURED DURING THE OPERATION. THE OPERATION WILL BE RETRIED 3 TIMES.

EM35 INVALID ADDRESS ('IAE') ERROR

AN INVALID ADDRESS ERROR OCCURED 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 OCCURED. 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 RH11/RP04 ERROR OCCURED 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 RP04 ERROR DETECTED

NO RP04 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 RP04 SET OR ERROR BITS IN THE RH11 SET.

EM50 SEEK INCOMPLETE OR OFF CYLINDER ERROR

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

EM51 PROGRAM DETECTED POSITIONING ERROR

A HEADER COMPARE ERROR OCCURED ('HCE'); HOWEVER, WHEN THE PROGRAM EXAMINED THE HEADER OF THE SECTOR IN ERROR, IT FOUND THAT THE CYLINDER FIELD DID NOT AGREE WITH THE CONTENTS OF 'RHCC' OF THE DRIVE. THE DRIVE WILL BE RECALIBRATED.

EM60 DEVICE UNSAFE

THE INDICATED DRIVE UNSAFE ERROR OCCURED; 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'

(DISPLAY OF THE RH11/RP04 REGISTERS IN TWO GROUPS: RHCS1,RHCS2,RHDS1,RHER1,RHER2,RHER3 RHEC1, & RHEC2 FORM THE FIRST GROUP OF REGISTERS;

ALL THE OTHER REGISTERS ARE IN THE SECOND GROUP.
IF SW<05> IS SET, ONLY THE REGISTERS IN THE FIRST
GROUP WILL BE DISPLAYED.)

MNEMONICS USED FOR THE DATA TRANSFER ORDERS ARE DEFINED BELOW:

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 7)
RDHD - READ HEADER & DATA (OCTAL 73)

'ERROR OCCURED DURING NON-DATA TRANSFER OPERATION'

THE ABOVE LINE WILL BE PRINTED IF THE ERROR OCCURED 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 RP04 REGISTERS. THE
CONTENTS OF THIS WORD IDENTIFY HOW THE ERROR WAS PROCESSED BY THE
RP04 DRIVE HANDLER ROUTINE. THE BITS IN THIS WORD ARE ENCODED
AS FOLLOWS:

BIT #	MEANING IF BIT IS '1'
15	ERROR OCCURED DONE (BIT07=0), BITS 14-9 SPECIFY TYPE DONE (BIT07=1), BITS 6-3 SPECIFY TYPE
14	DRIVE IS OFFLINE
12	PERSISTENT UNSAFE CONDITION EXISTS
11	UNCORRECTABLE ERROR OCCURED
10	FATAL PARITY ERROR OCCURED. 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 OCCURED DURING THE TRANSFER
5	ERROR OCCURED WHILE SEARCHING FOR THE 'TRANSFER' SECTOR
4	CORRECTABLE UNSAFE CONDITION OCCURED
3	DRIVE ERROR OCCURED THAT CAUSED AN AUTOMATIC

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

RHBA = XXXX RHWC = 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

RHDA = XXXX RHCA = YYYY

THIS LINE GIVES THE CONTENTS OF THE RP04 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

RHEC1 = XXXX RHEC2 = 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 OFFSET X MICRO-INCHES

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

LINE 16

ECC CORRECTABLE AT OFFSET X MICRO-INCHES

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 COUNT 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 OCCURED. IF THIS LINE IS PRINTED, THE RH11/RP04 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 'RHEC1' 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, OCYL 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 POS ERR' IS THE TOTAL NUMBER OF POSITIONING ERRORS WHICH THE PROGRAM DETECTED FOR THE DRIVE.

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

8. PROGRAM DESCRIPTION

8.1 PROGRAM OPERATION

WHEN THE PROGRAM IS STARTED FROM LOCATION 200(8), ALL TABLES AND PARAMETERS ARE CLEARED OR INITIALIZED. THE PARAMETERS WHICH ARE UNDER OPERATOR TTY ENTRY CONTROL ARE CHECKED FOR VALIDITY AND CONSISTENCY. RH11 INTERPUPT 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 RP04, 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 PERFORM THE OPERATION, THE PROGRAM ISSUES A SEARCH INSTURCTION 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 (RHLA) 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 NOTE 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 RPOH ERROR DETECTED - EM42
CAN'T MATCH DATA READ WITH A PATTERN - EM43
ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11 - EM44

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 MULTIDRIVE PROGRAM RECEIVES A REQUEST FOR A DRIVE, THE PROGRAM VERIFIES THAT THE DRIVE IS ONLINE. THE DRIVE IS SELECTED AND 0'S ARE WRITTEN INTO 'RHDS1': IF THE DRIVE IS IN NEUTRAL, THIS WILL SEIZE THE DRIVE. IF THE DRIVE IS SEIZED BY THE OTHER PORT, WRITING INTO 'RHDS1' WILL SET 'PORT REQUEST'. THE PROGRAM CHECKS 'DVA' IN 'RHCS1'. 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 R' PONSE TO PORT REQUEST' ERROR. NORMALLY THIS ERROR MESSAGE INDICATES A FAILURE IN THE DUAL PORT CONTROL LOGIC IN THE RP04 BEING TESTED; HOWEVER, UNDER CERTAIN CONDITIONS (E.G. MASSBUS PARITY ERRORS BEING REPORTED ON THE OTHER SYSTEM ON A M0033 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 THE RP04 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. NOTE THAT THIS CAN BE AN EXTENDED INTERVAL IN THE CASE OF A DRIVE UNSAFE WHICH CAUSES THE DRIVE TO CYCLE DOWN BUT WHICH IS CLEARED BY THE 'DRIVE CLEAR'.

SINGLE PORT DRIVES, DRIVES WHICH ARE LOCKED ON PORT, OR 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'.

B. THE BUFFER SIZE IS RANDOM SELECTED BETWEEN 4 (10) - AND THE VALUE IN 'MAXDL'. THE SIZE SELECTED IS WEIGHTED TO ENSURE THAT AT LEAST 4 WORDS ARE WRITTEN IN THE DATA AREA OF THE LAST SECTOR. 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 KEYWORDS IN THE HEADER (WHEN PERFORMING A WRITE HEADER & DATA ORDER) ARE ZERO FILLED. THE PROGRAM EXPECTS TO FIND THAT THE KEYWORDS ARE ZERO.

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 260 (10); THE PROGRAM WILL NOT PERFORM A MULTI-SECTOR FORMAT WRITE OPERATION.

E. THE FIRST ORDER PERFORMED AFTER A UNIT IS ASSIGNED WITH A 'T' 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. TO MAINTAIN COMPATIABILITY WITH PACKS WRITTEN BY THE FORMAT PROGRAM (MAINDEC-11-DZRPL), THE PROGRAM WILL ACCEPT ALL ZERO'S AND AND ALL ONE'S PATTERNS; HOWEVER, ALL ZERO'S AND ALL ONE'S PATTERNS ARE NOT WRITTEN BY THE EXERCISER PROGRAM.

PATTERN '8' IS DEFINED AS THE 'WORST CASE' PATTERN.

PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7	PAT 8
000001	177776	000000	000000	052525	007417	026455	165555
000003	177774	000000	010421	052525	007417	026455	133333
000007	177770	000000	021042	052525	007417	026455	165555
000017	177760	177777	031463	125252	170360	151322	133333
000037	177740	177777	042104	125252	170360	151322	165555
000077	177700	177777	052525	125252	170360	151322	133333
000177	177600	000000	063146	052525	007417	026455	165555
000377	177400	000000	073567	052525	007417	026455	133333
000777	177000	177777	104210	125252	170360	151322	165555
001777	176000	177777	114631	125252	170360	151322	133333
003777	174000	000000	125252	052525	007417	026455	165555
007777	170000	177777	135673	125252	170360	151322	133333
017777	160000	000000	146314	052525	007417	026455	165555
037777	140000	177777	156735	125252	170360	151322	133333
077777	100000	000000	167356	052525	007417	026455	165555
177777	000000	177777	177777	125252	170360	151322	133333

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

9. PROGRAM LISTING

3679	OPERATIONAL SWITCH SETTINGS
3693	BASIC DEFINITIONS
3697	RP04 DRIVER COMMANDS
3722	TRAP CATCHER
(1)	STARTING ADDRESS(ES)
3832	COMMON TAGS
(3)	CONTROL PARAMETERS
(3)	RP04 ADDRESS LIMIT VALUES
(3)	VALUES FOR FIRST OPERATION
(1)	ERROR POINTER TABLE
3886	SETUP AND INITIALIZATION ROUTINE
4121	MAIN PROGRAM
5868	ERROR MESSAGE GENERATION ROUTINES
6225	GENERAL SUPPORT SUBROUTINES
7213	TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
7238	TYPE ROUTINE
7286	PRINT ROUTINE
7334	TTY INPUT ROUTINE
7456	MACRO ROUTINES
7468	ERROR HANDLER ROUTINE
7469	ERROR MESSAGE TYPEOUT ROUTINE
7470	BINARY TO OCTAL (ASCII) AND TYPE
7471	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7472	SAVE AND RESTORE R0-R5 ROUTINES
7473	RANDOM NUMBER GENERATOR ROUTINE
7474	ROUTINE TO SIZE MEMORY
7475	DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
7476	SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
7477	DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
7478	SINGLE LENGTH BINARY TO OCTAL ASCIZ ROUTINE
7479	TRAP DECODER
(3)	TRAP TABLE
7485	RH11/RP04 DRIVER - SINGLE/DUAL PORT VERSION
8902	DATA, CONTROL, & STATUS BLOCKS
8990	TABLES, CONSTANTS, AND VARIABLE LOCATIONS
9089	DATA PATTERNS
9374	PRINTER/TELETYPE MESSAGES
9662	PATCH AREA

```
                                .TITLE  MAINDEC-11-DZRPN-B
                                ;*COPYRIGHT (C) 1974,1975
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY C. HESS
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-A2).
                                ;*

                                .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            PARTIAL REGISTER DISPLAY IF ERROR
                                ;*         5            NO ECC CORRECTION RESULTS DISPLAYED IF ERROR
                                ;*         4            DO NOT CHECK FOR MAXIMUM ERROR COUNTS
                                ;*         4              DO NOT DROP UNIT WHEN NORMAL END OF TEST REACHED
                                ;*         3            DISPLAY ERROR SECTOR IF 'DCK', 'DTE', OR 'WCF' ERROR
                                ;*         3              DISPLAY SECTOR IF 'DCK' ERR UNCORRECTABLE AFTER
                                ;*         3                  28TH RETRY
                                ;*         3              IF DATA COMPARE ERROR & SW7 SET, DISPLAY
                                ;*         3                  REMAINDER OF BUFFER
                                ;*         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
  177776                        PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
  177774                        STKLMT= 177774          ;;STACK LIMIT REGISTER
  177772                        PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
  177570                        SWR=    177570          ;;SWITCH REGISTER
  177570                        DISPLAY=SWR

                                ;*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
```

```
(1)  000007                  R7=     %7              ;;GENERAL REGISTER
(1)                          .EQUIV  R6,SP           ;;STACK POINTER
(1)                          .EQUIV  R7,PC           ;;PROGRAM COUNTER
(1)
(1)                          ;*PRIORITY LEVEL DEFINITIONS
(1)  000000                  PR0=    0               ;;PRIORITY LEVEL 0
(1)  000040                  PR1=    40              ;;PRIORITY LEVEL 1
(1)  000100                  PR2=    100             ;;PRIORITY LEVEL 2
(1)  000140                  PR3=    140             ;;PRIORITY LEVEL 3
(1)  000200                  PR4=    200             ;;PRIORITY LEVEL 4
(1)  000240                  PR5=    240             ;;PRIORITY LEVEL 5
(1)  000300                  PR6=    300             ;;PRIORITY LEVEL 6
(1)  000340                  PR7=    340             ;;PRIORITY LEVEL 7
(1)
(1)                          ;*"SWITCH REGISTER" SWITCH DEFINITIONS
(1)  100000                  SW15=   1 0000
(1)  040000                  SW14=   40000
(1)  020000                  SW13=   20000
(1)  010000                  SW12=   10000
(1)  004000                  SW11=   4000
(1)  002000                  SW10=   2000
(1)  001000                  SW09=   1000
(1)  000400                  SW08=   400
(1)  000200                  SW07=   200
(1)  000100                  SW06=   100
(1)  000040                  SW05=   40
(1)  000020                  SW04=   20
(1)  000010                  SW03=   10
(1)  000004                  SW02=   4
(1)  000002                  SW01=   2
(1)  000001                  SW00=   1
(1)                          .EQUIV  SW09,SW9
(1)                          .EQUIV  SW08,SW8
(1)                          .EQUIV  SW07,SW7
(1)                          .EQUIV  SW06,SW6
(1)                          .EQUIV  SW05,SW5
(1)                          .EQUIV  SW04,SW4
(1)                          .EQUIV  SW03,SW3
(1)                          .EQUIV  SW02,SW2
(1)                          .EQUIV  SW01,SW1
(1)                          .EQUIV  SW00,SW0
(1)
(1)                          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)  100000                  BIT15=  100000
(1)  040000                  BIT14=  40000
(1)  020000                  BIT13=  20000
(1)  010000                  BIT12=  10000
(1)  004000                  BIT11=  4000
(1)  002000                  BIT10=  2000
(1)  001000                  BIT09=  1000
(1)  000400                  BIT08=  400
(1)  000200                  BIT07=  200
(1)  000100                  BIT06=  100
(1)  000040                  BIT05=  40
```

```
(1)  000020  BIT04=  20
(1)  000010  BIT03=  10
(1)  000004  BIT02=  4
(1)  000002  BIT01=  2
(1)  000001  BIT00=  1
(1)          .EQUIV  BIT09,BIT9
(1)          .EQUIV  BIT08,BIT8
(1)          .EQUIV  BIT07,BIT7
(1)          .EQUIV  BIT06,BIT6
(1)          .EQUIV  BIT05,BIT5
(1)          .EQUIV  BIT04,BIT4
(1)          .EQUIV  BIT03,BIT3
(1)          .EQUIV  BIT02,BIT2
(1)          .EQUIV  BIT01,BIT1
(1)          .EQUIV  BIT00,BIT0
(1)
(1)          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)  000004  ERRVEC= 4              ;;TIME OUT AND OTHER ERRORS
(1)  000010  RESVEC= 10             ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)  000014  TBITVEC=14             ;;"T" BIT
(1)  000014  TRTVEC= 14             ;;TRACE TRAP
(1)  000014  BPTVEC= 14             ;;BREAKPOINT TRAP (BPT)
(1)  000020  IOTVEC= 20             ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)  000024  PWRVEC= 24             ;;POWER FAIL
(1)  000030  EMTVEC= 30             ;;EMULATOR TRAP (EMT) **ERROR**
(1)  000034  TRAPVEC=34             ;;"TRAP" TRAP
(1)  000060  TKVEC=  60             ;;TTY KEYBOARD VECTOR
(1)  000064  TPVEC=  64             ;;TTY PRINTER VECTOR
(1)  000240  PIRQVEC=240            ;;PROGRAM INTERRUPT REQUEST VECTOR
3694
3695         ;*****************************************************************
3696
3697         .SBTTL  RP04 DRIVER COMMANDS
3698
3699         ;*****************************************************************
3700
3701  000101  RNOP=   101            ;NO OPERATION
3702  000103  UNLOAD= 103            ;UNLOAD
3703  000105  SEEK=   105            ;SEEK
3704  000107  RECAL=  107            ;RECALIBRATE
3705  000111  DRVCLR= 111            ;DRIVE CLEAR
3706  000113  REL=    113            ;RELEASE
3707  000115  OFFSET= 115            ;OFFSET
3708  000117  RTC=    117            ;RETURN TO CENTER LINE
3709  000121  PRESET= 121            ;READ IN PRESET
3710  000123  ACK=    123            ;PACK ACKNOWLEDGE
3711  000131  SEARCH= 131            ;SEARCH
3712  000141  GETREG= 141            ;GET REGISTERS
3713  000143  SETFMT= 143            ;SET FORMAT (& ECI OR HCI)
3714  000145  SELDRV= 145            ;SELECT DRIVE
3715  000151  WCKD=   151            ;WRITE CHECK DATA
3716  000153  WCKHD=  153            ;WRITE CHECK HEADER & DATA
3717  000161  WRTDAT= 161            ;WRITE DATA
3718  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

                                        .SBTTL  STARTING ADDRESS(ES)
                000200                          .=200

 000200  000137  001430                         JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
 000204  000137  001430                         JMP     @#START         ;ADDRESS IS CHANGED TO 'RSTART' BY PROGRAM
                                                                         ;AFTER INITIAL START
```

```
3832                                    ;*****************************************************************
 (1)
 (1)                                    .SBTTL  COMMON TAGS
 (1)
 (1)                                    ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 (1)                                    ;*USED IN THE PROGRAM.
 (1)
 (1)            001100                          .=1100
 (1)
 (1)  001100                            $CMTAG:                 ;;START OF COMMON TAGS
 (1)  001100  000000                    $PASS:  .WORD   0       ;;CONTAINS PASS COUNT
 (1)  001102     000                    $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
 (1)  001103     000                    $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
 (1)  001104  000000                    $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
 (1)  001106  000000                    $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP
 (1)  001110  000000                    $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
 (1)  001112  000000                    $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
 (1)  001114     000                    $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
 (1)  001115     001                    $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
 (1)  001116  000000                    $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
 (1)  001120  000000                    $GDADR: .WORD   0       ;;CONTAINS  OF 'GOOD' DATA
 (1)  001122  000000                    $BDADR: .WORD   0       ;;CONTAINS  OF 'BAD' DATA
 (1)  001124  000000                    $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
 (1)  001126  000000                    $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
 (1)  001130  000000  000000  000000            .WORD   0,0,0   ;;RESERVED--NOT TO BE USED
 (1)  001136  177560                    $TKS:   177560          ;;TTY KBD STATUS
 (1)  001140  177562                    $TKB:   177562          ;;TTY KBD BUFFER
 (1)  001142  177564                    $TPS:   177564          ;;TTY PRINTER STATUS REG.
 (1)  001144  177566                    $TPB:   177566          ;;TTY PRINTER BUFFER REG.
 (1)  001146     000                    $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
 (1)  001147     002                    $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 (1)  001150     012                    $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
 (1)  001151     000                    $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 (1)  001152  177607  000377            $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
 (1)  001156     077                    $QUES:  .ASCII  /?/     ;;QUESTION MARK
 (1)  001157     015                    $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
 (1)  001160  000012                    $LF:    .ASCIZ  <12>    ;;LINE FEED
 (3)  001162  176700                    $RPADR: .WORD   176700  ;FIRST ADDRESS OF RH11/RP04 REGISTERS
 (3)  001164  000254                    $RPVEC: .WORD   254     ;RP04 VECTOR ADDRESS
 (3)  001166  172540                    $LKCSR: .WORD   172540  ;ADDR OF KW11-P STATUS REGISTER
 (3)  001170  172542                    $LKCSB: .WORD   172542  ;ADDR OF KW11-P COUNTER BUFFER
 (3)  001172  000104                    $LPVEC: .WORD   104     ;ADDR OF KW11-P VECTOR
 (3)  001174  177546                    $LKS:   .WORD   177546  ;ADDR OF KW11-L STATUS REGISTER
 (3)  001176  000100                    $LLVEC: .WORD   100     ;ADDR OF KW11-L VECTOR
 (3)  001200  177777                    PCLOCK: .WORD   -1      ;'0' IF KW11-P IS ON SYSTEM
 (3)  001202  177777                    CLKFLG: .WORD   -1      ;'0' IF A CLOCK IS AVAILABLE
 (3)  001204  177777                    $PRFLG: .WORD   -1      ;PRINTER AVAILABILITY FLAG
 (3)  001206  177514                    $LSCS:  .WORD   177514  ;PRINTER STATUS WORD ADDRESS
 (3)  001210  177516                    $LSDB:  .WORD   177516  ;PRINTER DATA BUFFER ADDRESS
 (3)  001212  000000                    $PRTER: .WORD   0       ;PRINTER ERROR FLAG
 (3)  001214  000074                    HZ:     .WORD   74      ;74 (8) IF 60 HZ SYSTEM; 62 (8) IF 50 HZ SYSTEM
 (3)  001216  000000                    LA30:   .WORD   0       ;0 IF LA30, 1'S IF MOD 33 OR MOD 35 TTY ON SYSTEM
 (3)  001220  001100                    STACK1: .WORD   STACK   ;CONSTANT USED TO CHECK STACK POINTER
 (3)  001222  000000                    STATIN: .WORD   0       ;ERROR RATE DATA DISPLAY INDICATOR
```

```
(3)  001224  000000                    PACK:   .WORD   0               ;'W ' COMMAND INDICATOR
(3)  001226  000000  000000  000000    DATE:   .WORD   0,0,0,0,0       ;OPERATOR ENTERED DATE
(3)  001234  000000  000000
(3)  001240  000000  000000  000000    OPERID: .WORD   0,0,0,0         ;OPERATOR ID
(3)  001246  000000
(3)  001250  000000                    DRIVE:  .WORD   0               ;DRIVE # STORAGE: ERRORS 1-5 & 10
(3)  001252  000000                    ATTN:   .WORD   0               ;ATTN REG STORAGE: ERRORS 1-5 & 10
(3)  001254     040     060     000    UNIT:   .BYTE   40,60,0         ;DRIVE # STORAGE FOR PRINTOUT
(3)          001260                             .EVEN
(3)
(4)                                     ;*****************************************************************
(3)
(3)                                     .SBTTL  CONTROL PARAMETERS
(3)
(4)                                     ;*****************************************************************
(3)
(3)  001260  002740                    ENDCON: .WORD   002740          ;1.875X10↑8 WORDS (10) [3X10↑9 BITS]
(3)  001262  005455                            .WORD   005455          ;MSW
(3)  001264  143300                    ENDSEK: .WORD   143300          ;3 X 10↑6 SEEKS  (LSW)
(3)  001266  000055                            .WORD   55              ;MSW
(3)  001270  000001                    PASCNT: .WORD   1               ;NUMBER OF PASSES TO END OF TEST
(3)  001272  000000                    MAXDL:  .WORD   0               ;MAXIMUM DATA TRANFER SIZE IN WORDS
(3)                                                                    ;(FILLED BY PROGRAM AT STARTUP OR BY OPERATOR
(3)                                                                    ;DURING PARAMETER ENTRY DIALOG.)
(3)  001274  000144                    MAXER:  .WORD   100.            ;MAXIMUM ERRORS - 100(10)
(3)  001276  000005                    INTRVL: .WORD   5               ;LOW BYTE IS PERFORMANCE TYPEOUT INTERVAL
(3)                                                                    ;COUNTER.  UPPER BYTE IS VALUE.
(3)  001300  000004                    CMPLMT: .WORD   4               ;NUMBER OF COMPARE ERRORS TYPED OUT
(3)  001302  000001                    FORMAT: .WORD   1               ;IF EQ 1, ALLOW WRITE HEADER & DATA ORDERS
(3)                                                                    ;IF EQ 0, DO NOT ALLOW WRITE HEADER & DATA ORDERS
(3)  001304  000001                    SAVLOD: .WORD   1               ;IF EQ 1, SAVE THE LOADER
(3)                                                                    ;IF EQ 0, DO NOT SAVE THE LOADER
(3)  001306  000003                    RATIO:  .WORD   3               ;READ/WRITE RATIO [RANGE 0 - 7]
(3)                                                                    ;0 - 15/1       (READ/WRITE)
(3)                                                                    ;1 - 7/1
(3)                                                                    ;2 - 6/2
(3)                                                                    ;3 - 5/3
(3)                                                                    ;4 - 4/4
(3)                                                                    ;5 - 3/5
(3)                                                                    ;6 - 2/6
(3)                                                                    ;7 - 1/7
(3)  001310  000001                    AUTOCK: .WORD   1               ;IF EQ 1, DO AN APPROPRITE WRITE
(3)                                                                    ;CHECK AFTER EACH WRITE ORDER.
(3)                                                                    ;IF EQ 0, SELECT WRITE CHECK ORDERS
(3)                                                                    ;RANDOMLY.
(3)  001312  000001                    NOTPRT: .WORD   1               ;IF EQ 1, DO NOT PRINT DATA ERROR MESSAGES
(3)                                                                    ;ASSOCIATED WITH OPERATOR SPECIFIED
(3)                                                                    ;BAD PACK AREAS.
(3)                                                                    ;IF EQ 1, PRINT ERROR MESSAGES RELATING TO
(3)                                                                    ;THESE AREAS.
(3)  001314  000001                    ENDET:  .WORD   1               ;IF EQ 1, END OF PASS DETERMINED
(3)                                                                    ;BY THE 'WORDS READ' COUNT.
(3)                                                                    ;IF EQ 0, END OF PASS DETERMINED
(3)                                                                    ;BY THE SEEK COUNT.
```

```
(3)
(4)                             ;*****************************************************************
(3)
(3)                             .SBTTL  RP04 ADDRESS LIMIT VALUES
(3)
(4)                             ;*****************************************************************
(3)
(3)  001316  000632             MAXCYL: .WORD   632             ;MAX CYLINDER [RANGE 0 - 632 (OCTAL)]
(3)  001320  000000             MINCYL: .WORD   0               ;MINIMUM CYLINDER [RANGE 0 - (= MAXCYL]
(3)  001322  000022             MAXTRK: .WORD   22              ;MAX TRACK [RANGE 0 - 22 (OCTAL)]
(3)  001324  000000             MINTRK: .WORD   0               ;MIN TRACK [RANGE 0 TO (= MAXTRK]
(3)  001326  000025             MAXSEC: .WORD   25              ;MAX SECTOR [RANGE 0 - 25 (OCTAL)]
(3)  001330  000000             MINSEC: .WORD   0               ;MIN SECTOR [RANGE 0 TO (= MAXSEC]
(3)
(4)                             ;*****************************************************************
(3)
(3)                             .SBTTL  VALUES FOR FIRST OPERATION
(3)
(4)                             ;*****************************************************************
(3)
(3)  001332  000010             BEGPAT: .WORD   10              ;STARTING PATTERN CODE [RANGE 1 - 17 (OCTAL)]
(3)  001334  000005             BEGCOD: .WORD   5               ;STARTING COMMAND CODE [RANGE 0 - 5]
(3)                                                             ;0 = WRITE CHECK DATA ('WCKD')
(3)                                                             ;1 = WRITE CHECK HEADER & DATA ('WCHKHD')
(3)                                                             ;2 = WRITE DATA ('WRTDAT')
(3)                                                             ;3 = WRITE HEADER & DATA ('WRTHD')
(3)                                                             ;4 = READ DATA ('RDDAT')
(3)                                                             ;5 = READ HEADER & DATA ('RDHD')
(3)  001336  000404             BEGSIZ: .WORD   404             ;STARTING RECORD SIZE [RANGE 4 - MAXMEM]
(3)                                                             ;NOTE:  THE SIZE MUST BE AT LEAST 4 IF
(3)                                                             ;WRITE DATA OR READ DATA; THE SIZE MUST
(3)                                                             ;BE AT LEAST 8 IF WRITE HEADER AND
(3)                                                             ;DATA OR READ HEADER AND DATA.
(3)                                                             ;IF THE SIZE IS GREATER THAN 1 SECTOR, THE
(3)                                                             ;SIZE MUST ALLOW FOR OVERLAPPING 4 OR 8
(3)                                                             ;WORDS INTO THE LAST SECTOR USED.
```

```
    (2)                             ;*****************************************************************
    (1)
    (1)                             .SBTTL  ERROR POINTER TABLE
    (1)
    (1)                             ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
    (1)                             ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
    (1)                             ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
    (1)                             ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
    (1)                             ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
    (1)
    (1)                             ;*      EM              ;;POINTS TO THE ERROR MESSAGE
    (1)                             ;*      DH              ;;POINTS TO THE DATA HEADER
    (1)                             ;*      DT              ;;POINTS TO THE DATA
    (1)                             ;*      DF              ;;POINTS TO THE DATA FORMAT
    (1)
    (1)
    (1)  001340                     $ERRTB:
 3833                               ;ERROR 1
 3834
 3835  001340  043622                       EM1                     ;ILLEGAL RH11 INTERRUPT
 3836  001342  000000                       0
 3837  001344  000000                       0
 3838  001346  000000                       0
 3839
 3840                               ;ERROR 2
 3841
 3842  001350  043672                       EM2                     ;UNEXPECTED ATTENTION DETECTED
 3843  001352  046003                       DH2                     ;DRV RHAS
 3844  001354  046422                       DT2                     ;DATA POINTER
 3845  001356  047262                       DF2                     ;DRV-DEC, REG=OCTAL
 3846
 3847                               ;ERROR 3
 3848
 3849  001360  043717                       EM3                     ;MASSBUS CONTROL BUS PARITY ERROR (SEEN BY THE RH11)
 3850  001362  046060                       DH3                     ;DRV ADDR OF REG CONTENTS
 3851  001364  046440                       DT3                     ;DATA POINTER
 3852  001366  047270                       DF3                     ;ALL OCTAL EXCEPT DRV
 3853
 3854                               ;ERROR 4
 3855
 3856  001370  043775                       EM4                     ;CONTROL BUS PARITY ERROR (SEEN BY THE RP04)
 3857  001372  046106                       DH4                     ;DRV REG ADDR WRITTEN READ
 3858  001374  046450                       DT4                     ;DATA POINTER
 3859  001376  047273                       DF4                     ;OCTAL EXCEPT DRV
 3860
 3861                               ;ERROR 5
 3862
 3863  001400  044053                       EM5                     ;ATTEN FROM AN OFFLINE OR UNAVAIL DRIVE
 3864  001402  046003                       DH2                     ;DRV RHAS
 3865  001404  046422                       DT2                     ;DATA POINTER
 3866  001406  047262                       DF2                     ;OCTAL
 3867
 3868                               ;ERROR 6
 3869
```

```
 3870  001410  000000                          0                       ;RESERVED
 3871  001412  000000                          0
 3872  001414  000000                          0
 3873  001416  000000                          0
 3874
 3875                                  ;ERROR 7
 3876
 3877  001420  000000                          0                       ;RESERVED
 3878  001422  000000                          0
 3879  001424  000000                          0
 3880  001426  000000                          0
 3881
 3882
 3884                                  ;*****************************************************************
 3885
 3886                                  .SBTTL  SETUP AND INITIALIZATION ROUTINE
 3887
 3888                                  ;       START ADDRESS = 200
 3889                                  ;       RESTART ADDRESS = 204
 3890
 3891                                  ;*****************************************************************
 3892
 3893  001430                          START:
  (1)  001430  012737  000340  177776          MOV     #340,@#PS       ;;LOCK OUT ALL INTERRUPTS
  (1)  001436  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  001442  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  001444  022706  001136                  CMP     #$TKS,R6        ;;DONE?
  (1)  001450  001374                          BNE     .-6             ;;LOOP BACK IF NO
  (1)  001452  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)  001456  012737  026514  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001464  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  001472  012737  030370  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  001500  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 3894  001506  104400  052010          1$:     TYPE    ,TITLE          ;TYPE THE PROGRAM NAME AND MAINDEC NUMBER
 3895  001512  012737  001157  001510          MOV     #$CRLF,1$+2     ;CHANGE 'TITLE' TO CR-LF
 3896  001520  012737  000240  000032          MOV     #240,@#EMTVEC+2   ;CHANGE EMT PRIORITY TO 5
 3897  001526  012737  000240  000036          MOV     #240,@#TRAPVEC+2  ;CHANGE TRAP PRIORITY TO 5
 3898  001534  012737  002214  000206          MOV     #RSTART,206     ;SETUP RESTART ADDRESS
 3899  001542  005037  001222                  CLR     STATIN          ;CLEAR ERROR RATE DATA TYPE INDICATOR
 3900  001546  105037  001151                  CLRB    $TPFLG          ;SET TTY AVAILABILITY FLAG
 3901  001552  012705  041526                  MOV     #ORDERQ,R5      ;START OF AREA TO CLEAR
 3902  001556  005025                          CLR     (R5)+
 3903  001560  022705  042034                  CMP     #BLKADR,R5      ;LOOK FOR END OF CLEAR AREA
 3904  001564  001374                          BNE     .-6             ;BR IF NOT FINISHED
 3905  001566  013737  001214  021632          MOV     HZ,SIXTEE       ;1/60 TH OR 1/50 TH SECOND COUNTER VALUE
 3906  001574  005437  021632                  NEG     SIXTEE          ;CONVERT TO 2'S COMP
 3907  001600  012737  030060  021614          MOV     #30060,HOUR     ;ASCII '00' TO HOUR COUNTER
 3908  001606  012737  030060  021620          MOV     #30060,MINUTE   ;ASCII '00' TO MINUTES COUNTER
 3909  001614  012737  030060  021624          MOV     #30060,SECOND   ;ASCII '00' TO SECONDS COUNTER
 3910  001622  105037  001277                  CLRB    INTRVL+1        ;CLEAR INTERVAL COUNTER
 3911  001626  104400  052337                  TYPE    ,ENTDAT         ;'ENTER DATE'
 3912  001632  104414                          RDLIN                   ;READ THE ENTRY
 3913  001634  012605                          MOV     (SP)+,R5        ;PUT THE ENTRY ADDRESS INTO R5
 3914  001636  012537  001226                  MOV     (R5)+,DATE      ;STORE THE DATE
```

```
3915  001642  012537  001230                MOV     (RS)+,DATE+2    ;STORE THE DATE
3916  001646  012537  001232                MOV     (RS)+,DATE+4    ;STORE THE DATE
3917  001652  012537  001234                MOV     (RS)+,DATE+6    ;STORE THE DATE
3918  001656  104400  052356                TYPE    ,ENTID          ;'ENTER OPERATOR I.D.'
3919  001662  104414                       RDLIN                   ;READ THE ENTRY
3920  001664  012605                       MOV     (SP)+,R5        ;ENTRY ADDRESS
3921  001666  012537  001240                MOV     (R5)+,OPERID    ;STORE THE I.D.
3922  001672  012537  001242                MOV     (R5)+,OPERID+2  ;STORE THE I.D.
3923  001676  012537  001244                MOV     (R5)+,OPERID+4  ;STORE THE I.D.
3924
3925                                   ;ROUTINE TO DETERMINE BUFFER AREA SIZE
3926
3927  001702  004737  027700        SIZMEM: JSR     PC,$SIZE        ;SEE HOW MUCH MEMORY ON SYSTEM
3928  001706  022737  053564  027772        CMP     #ENDPGM,$LSTAD  ;SEE IF ENOUGH MEMORY FOR PROGRAM
3929  001714  103402                       BLO     1$              ;BR IF MEMORY
3930  001716  000137  020014                JMP     MEMERR          ;REPORT NOT ENOUGH MEMORY
3931  001722  012737  000001  041712  1$:   MOV     #1,BUFTBL       ;LOAD NUMBER OF BUFFERS
3932  001730  012737  053564  041714        MOV     #ENDPGM,BUFTBL+2  ;STARTING ADDRESS OF BUFFER
3933  001736  013737  027772  041716        MOV     $LSTAD,BUFTBL+4 ;LAST ADDR TO BUFFER ALLOCATION TABLE
3934  001744  162737  053564  041716        SUB     #ENDPGM,BUFTBL+4  ;SUBTRACT PROGRAM SPACE
3935  001752  000241                       CLC                     ;CLEAR THE 'C' BIT
3936  001754  006037  041716                ROR     BUFTBL+4        ;CONVERT TO WORD COUNT
3937  001760  105737  000041                TSTB    41              ;SEE WHO LOADED THE PROGRAM
3938  001764  001004                       BNE     2$              ;BR IF LOADED BY 'XXDP'
3939  001766  162737  000140  041716        SUB     #96.,BUFTBL+4   ;SUBTRACT 'A&S' LOADED SIZE
3940  001774  000403                       BR      3$              ;CONTINUE WITH BUFFER SPACE SETUP
3941  001776  162737  002002  041716  2$:   SUB     #1026.,BUFTBL+4   ;SUBTRACT 'XXDP' LOADER SIZE
3942  002004  005737  001272        3$:   TST     MAXDL           ;VALUE IN 'MAXDL' ?
3943  002010  001012                       BNE     LKPAR           ;BR IF VALUE IS
3944  002012  012737  013534  001272        MOV     #5980.,MAXDL    ;ASSUME FULL TRACK + 1 SEC MAXIMUM
3945  002020  023737  001272  041716        CMP     MAXDL,BUFTBL+4  ;IS THAT TOO LARGE ?
3946  002026  103403                       BLO     LKPAR           ;BR IF NOT
3947  002030  013737  041716  001272        MOV     BUFTBL+4,MAXDL  ;USE MAX AVAIL MEMORY AS MAX BUFFER SIZE
3948
3949                                   ;SEE IF THE OPERATOR WANTS TO CHANGE ANY PARAMETERS
3950
3951  002036  104400  052772        LKPAR:  TYPE    ,ASKPAR         ;ASK FOR PARMETERS
3952  002042  104414                       RDLIN                   ;READ THE ENTRY
3953  002044  012605                       MOV     (SP)+,R5        ;ADDRESS OF ENTRY TO R5
3954  002046  122715  000131                CMPB    #'Y,(R5)        ;WAS ENTRY A 'Y' (YES)
3955  002052  001404                       BEQ     ENTPAR          ;BR IF A 'Y'
3956  002054  122715  000116                CMPB    #'N,(R5)        ;WAS ENTRY A 'N' (NO)
3957  002060  001455                       BEQ     RSTART          ;BR IF AN 'N'
3958  002062  000765                       BR      LKPAR           ;NOT A VALID RESPONSE - RETRY
3959
3960                                   ;PARAMETER ENTRY ROUTINE
3961
3962  002064  104400  001157        ENTPAR: TYPE    ,$CRLF          ;CR-LF
3963  002070  005003                       CLR     R3              ;CLEAR OCTAL PARAMETER INDICATOR
3964  002072  012704  052646                MOV     #PARLST,R4      ;ENTRY PARAMETER LIST TO R4
3965  002076  012437  002114        1$:   MOV     (R4)+,3$        ;ADDRESS OF PARAMETER NAME
3966  002102  001427                       BEQ     ENTPR           ;BR IF AT END OF TABLE
3967  002104  100002                       BPL     2$              ;BR IF NOT DECIMAL-OCTAL SEPARATOR
3968  002106  005203                       INC     R3              ;SET OCTAL PARAMETER INDICATOR
```

```
3969  002110  000772                        BR      1$              ;CONTINUE WITH PARAMETER REQUESTS
3970  002112  104400                2$:     TYPE                    ;TYPE THE PARAMETER NAME
3971  002114  000000                3$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
3972  002116  012405                        MOV     (R4)+,R5        ;ADDRESS OF PARAMETER LOCATION
3973  002120  011546                        MOV     (R5),-(SP)      ;CONTENTS OF PARAMETER LOCATION
3974  002122  005703                        TST     R3              ;SEE IF OCTAL OR DECIMAL PARAMETER
3975  002124  001007                        BNE     4$              ;BR IF OCTAL
3976  002126  104410                        TYPOS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
3977  002130  104400  053105                TYPE    ,DEC10          ;TYPE DECIMAL INDICATOR
3978  002134  004737  025354                JSR     PC,GETDEC       ;GET THE DECIMAL ENTRY
3979  002140  012615                        MOV     (SP)+,(R5)      ;MOVE NEW VALUE TO LOCATION
3980  002142  000755                        BR      1$              ;GET MORE PARAMETERS
3981  002144  104402                4$:     TYPOC                   ;TYPE THE CURRENT VALUE
3982  002146  104400  053076                TYPE    ,OCT8           ;TYPE OCTAL INDICATOR
3983  002152  004737  025216                JSR     PC,GETOCT       ;GET THE OCTAL PARAMTER
3984  002156  012615                        MOV     (SP)+,(R5)      ;MOVE NEW VALUE TO PARAMETER LOCATION
3985  002160  000746                        BR      1$              ;GET MORE PARAMETERS
3986  002162  005737  001304        ENTPR:  TST     SAVLOD          ;SAVE THE LOADER ?
3987  002166  001012                        BNE     RSTART          ;BR IF LOADER TO BE SAVED
3988  002170  105737  000041                TSTB    41              ;SEE WHO LOADED THE PROGRAM
3989  002174  001004                        BNE     1$              ;BR IF 'ACT' OR 'XXDP'
3990  002176  062737  000140  041716        ADD     #96.,BUFTBL+4   ;'ABS' LOADER SIZE
3991  002204  000403                        BR      RSTART          ;CONTINUE
3992  002206  062737  002002  041716  1$:   ADD     #1026.,BUFTBL+4 ;'XXDP' LOADER SIZE
3993
3994                                ;RESTART (START FROM LOC 204) ENTERS HERE
3995
3996  002214  012706  001100        RSTART: MOV     #STACK,SP       ;RESTORE STACK POINTER FOR RESTART
3997  002220  005037  177776                CLR     @#PS            ;CLEAR PROCESSOR STATUS
3998  002224  005037  001224                CLR     PACK            ;'R' OR 'W' COMMAND INDICATOR
3999  002230  012737  123456  027676        MOV     #123456,$LONUM  ;INITIALIZE LOW RANDOM NUMBER
4000  002236  012737  176543  027674        MOV     #176543,$HINUM  ;INITIALIZE HIGH RANDOM NUMBER
4001  002244  113737  001276  001277        MOVB    INTRVL,INTRVL+1 ;RESTORE STATISTICS TYPEOUT INTERVAL
4002  002252  042737  170000  001274        BIC     #170000,MAXER   ;MAKE SURE LIMITS ARE NOT TOO LARGE
4003
4004                                ;CHECK THE PARAMETERS FOR VALIDITY
4005
4006  002260  023737  001272  041716  CKADR: CMP     MAXDL,BUFTBL+4  ;REQUESTED TRANSFER SIZE TOO LARGE ?
4007  002266  101004                        BHI     1$              ;BR IF REQUESTED MAXIMUM TOO LARGE
4008  002270  022737  000004  001272        CMP     #4,MAXDL        ;SEE IF MAXIMUM TRANSFER SIZE TOO SMALL
4009  002276  101404                        BLOS    2$              ;BR IF NOT TOO SMALL
4010  002300  012746  053114        1$:     MOV     #PAR1,-(SP)     ;ADDR OF PARAMETER NAME
4011  002304  000137  002622                JMP     BADPAR          ;REPORT THE PARAMETER ERROR
4012  002310  023737  001272  001336  2$:   CMP     MAXDL,BEGSIZ    ;IS MAX RECORD LENGTH SMALLER THAN 1 SECTOR ?
4013  002316  103003                        BHIS    CKPAR           ;BR IF NOT
4014  002320  013737  001272  001336        MOV     MAXDL,BEGSIZ    ;USE MAX RECORD SIZE AS STARTING SIZE
4015  002326  022737  000632  001316  CKPAR: CMP     #410.,MAXCYL    ;MAXIMUM CYLINDER ADDR TOO LARGE
4016  002334  103004                        BHIS    1$              ;BR IF NOT
4017  002336  012746  053144                MOV     #PAR4,-(SP)     ;ADDR OF PARAMETER NAME
4018  002342  000137  002622                JMP     BADPAR          ;REPORT PARAMETER ERROR
4019  002346  023737  001316  001320  1$:   CMP     MAXCYL,MINCYL   ;MAX CYLINDER LARGER THAN MIN CYLINDER
4020  002354  103004                        BHIS    2$              ;BR IF IT IS
4021  002356  012746  053154                MOV     #PAR5,-(SP)     ;ADDR OF PARAMETER NAME
4022  002362  000137  002622                JMP     BADPAR          ;REPORT PARAMETER ERROR
```

```
 4023  002366  022737  000022  001322  2$:     CMP     #18.,MAXTRK     ;MAXIMUM TRACK ADDRESS TOO LARGE
 4024  002374  103004                          BHIS    3$              ;BR IF NOT
 4025  002376  012746  053164                  MOV     #PAR6,-(SP)     ;ADDRESS OF PARAMETER NAME
 4026  002402  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4027  002406  023737  001322  001324  3$:     CMP     MAXTRK,MINTRK   ;SEE IF MAX TRACK EQ OR LARGER THAN MIN TRACK
 4028  002414  103004                          BHIS    4$              ;BR IF IT IS
 4029  002416  012746  053174                  MOV     #PAR7,-(SP)     ;ADDRESS OF PARAMETER NAME
 4030  002422  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4031  002426  022737  000025  001326  4$:     CMP     #21.,MAXSEC     ;SEE IF MAX SECTOR TOO LARGE
 4032  002434  103004                          BHIS    5$              ;BR IF MAX SECTOR OK
 4033  002436  012746  053204                  MOV     #PAR8,-(SP)     ;ADDRESS OF PARAMETER NAME
 4034  002442  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4035  002446  023737  001326  001330  5$:     CMP     MAXSEC,MINSEC   ;MINIMUM SECTOR ADDRESS TOO LARGE
 4036  002454  103004                          BHIS    6$              ;BR IF NOT
 4037  002456  012746  053214                  MOV     #PAR9,-(SP)     ;ADDRESS OF PARAMETER NAME
 4038  002462  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4039  002466  022737  000007  001306  6$:     CMP     #7,RATIO        ;SEE IF R/W RATIO TOO LARGE
 4040  002474  103004                          BHIS    7$              ;BR IF NOT
 4041  002476  012746  053264                  MOV     #PAR14,-(SP)    ;ADDRESS OF PARAMETER NAME
 4042  002502  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4043  002506  022737  000017  001332  7$:     CMP     #15.,BEGPAT     ;SEE IF VALID PATTERN SELECTED
 4044  002514  103004                          BHIS    8$              ;BR IF PATTERN CODE VALID
 4045  002516  012746  053324                  MOV     #PAR18,-(SP)    ;ADDRESS OF PARAMETER NAME
 4046  002522  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4047  002526  005737  001334          8$:     TST     BEGCOD          ;IS STARTING CODE OK
 4048  002532  001447                          BEQ     SETVEC          ;BR IF OK
 4052  002534  022737  000001  001334          CMP     #1,BEGCOD       ;IS STARTING CODE OK ?
  (1)  002542  001443                          BEQ     SETVEC          ;BR IF OK
  (1)  002544  022737  000002  001334          CMP     #2,BEGCOD       ;IS STARTING CODE OK ?
  (1)  002552  001437                          BEQ     SETVEC          ;BR IF OK
  (1)  002554  022737  000004  001334          CMP     #4,BEGCOD       ;IS STARTING CODE OK ?
  (1)  002562  001433                          BEQ     SETVEC          ;BR IF OK
  (1)  002564  022737  000005  001334          CMP     #5,BEGCOD       ;IS STARTING CODE OK ?
  (1)  002572  001427                          BEQ     SETVEC          ;BR IF OK
 4053  002574  022737  000003  001334          CMP     #3,BEGCOD       ;IS STARTING CODE WRITE HEADER & DATA ?
 4054  002602  001003                          BNE     9$              ;BR IF NOT
 4055  002604  005737  001302                  TST     FORMAT          ;FORMAT ORDERS ALLOWED ?
 4056  002610  001020                          BNE     SETVEC          ;BR IF THEY ARE
 4057  002612  012746  053314          9$:     MOV     #PAR17,-(SP)    ;ADDRESS OF PARAMETER NAME
 4058  002616  000137  002622                  JMP     BADPAR          ;REPORT PARAMETER ERROR
 4059
 4060                                  ;REPORT THE BAD PARAMETER
 4061
 4062  002622  012637  002634  BADPAR: MOV     (SP)+,1$        ;MOVE PARAMETER NAME
 4063  002626  104400  053016                  TYPE    ,NGPAR          ;'BAD PARAMETER'
 4064  002632  104400                          TYPE
 4065  002634  000000          1$:     .WORD   0               ;PARAMETER NAME
 4066  002636  104400  053046                  TYPE    ,REPAR          ;'REENTER PARAMETERS
 4067  002642  105037  001277                  CLRB    INTRVL+1        ;CLEAR PERFORMANCE TYPEOUT INTERVAL COUNTER
 4068  002646  000137  002064                  JMP     ENTPAR          ;GO THROUGH PARAMETER ENTRY ROUTINE AGAIN
 4069
 4070                                  ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
 4071                                  ;       PROGRAM WILL USE
 4072
```

```
4073  002652  004737  020124          SETVEC: JSR     PC,CKPTR        ;SEE IF PRINTER ON SYSTEM
4074  002656  013737  001162  030624          MOV     $RPADR,RPADR    ;RP04 ADDRESS
4075  002664  013737  001164  030626          MOV     $RPVEC,RPVEC    ;RP04 VECTOR ADDRESS
4076  002672  004737  020172                  JSR     PC,CKCLK        ;SEE IF CLOCK ON SYSTEM
4077  002676  004737  030642                  JSR     PC,RPINIT       ;INITIALIZE THE RP04 DRIVER
4078  002702  012737  177777  030564          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
4079  002710  142737  000007  001255          BICB    #7,UNIT+1       ;CLEAR UNIT NUMBER
4080  002716  005004                          CLR     R4              ;DRIVE TABLE POINTER
4081  002720  012703  000010                  MOV     #8.,R3          ;ITERATION COUNTER
4082  002724  104400  001157                  TYPE    ,$CRLF          ;CR-LF
4083  002730  104400  051437                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
4084  002734  104400  001254          1$:     TYPE    ,UNIT           ;TYPE UNIT NUMBER
4085  002740  104400  051272                  TYPE    ,LIN4SP         ;SPACES
4086  002744  105764  030466                  TSTB    DRVSTA(R4)      ;SEE WHAT DRIVE'S STATUS IS
4087  002750  001405                          BEQ     2$              ;BR IF OFFLINE
4088  002752  100410                          BMI     3$              ;BR IF UNSAFE OR NONEXISTANT
4089  002754  012737  051315  003056          MOV     #UNTON,8$       ;MESSAGE ADDRESS
4090  002762  000434                          BR      7$              ;PRINT IT
4091  002764  012737  051304  003056  2$:     MOV     #UNTOFF,8$      ;OFFLINE MESSAGE ADDRESS
4092  002772  000430                          BR      7$              ;PRINT IT
4093  002774  006304                  3$:     ASL     R4              ;SETUP TO ADDRESS WORDS
4094  002776  005764  030476                  TST     DRVTYP(R4)      ;SEE IF UNIT PRESENT
4095  003002  001414                          BEQ     4$              ;BR IF NOT PRESENT
4096  003004  022764  020020  030476          CMP     #20020,DRVTYP(R4)  ;SEE IF SINLE PORT RP04
4097  003012  001414                          BEQ     5$              ;BR IF IT IS
4098  003014  022764  024020  030476          CMP     #24020,DRVTYP(R4)  ;SEE IF DUAL PORT RP04
4099  003022  001410                          BEQ     5$              ;BR IF IT IS
4100  003024  012737  051375  003056          MOV     #NOTRP,8$       ;UNIT IS NOT AN RP04
4101  003032  000407                          BR      6$              ;PRINT IT
4102  003034  012737  051412  003056  4$:     MOV     #NOTPRS,8$      ;UNIT NOT PRESENT
4103  003042  000403                          BR      6$              ;PRINT IT
4104  003044  012737  051427  003056  5$:     MOV     #NOTSAF,8$      ;UNIT UNSAFE
4105  003052  006204                  6$:     ASR     R4              ;RESTORE R4
4106  003054  104400                  7$:     TYPE                    ;TYPE THE IDENTIFYING MESSAGE
4107  003056  000000                  8$:     .WORD   0               ;MESSAGE ADDRESS HERE
4108  003060  104400  001157                  TYPE    ,$CRLF          ;CR-LF
4109  003064  005303                          DEC     R3              ;DECREMENT THE ITERATION COUNTER
4110  003066  001404                          BEQ     9$              ;BR IF END
4111  003070  105237  001255                  INCB    UNIT+1          ;INCREMENT THE UNIT NUMBER
4112  003074  005204                          INC     R4              ;INCREMENT TABLE POINTER
4113  003076  000716                          BR      1$              ;CONTINUE
4114  003100  104400  001157          9$:     TYPE    ,$CRLF          ;CR-LF
4115  003104  004737  026000                  JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD INTERRUPT HANDLER
4116  003110  104400  052600                  TYPE    ,INTDON         ;TYPE 'INITIALIZE COMPLETE'
4117  003114  000137  003120                  JMP     MAIN            ;START THE PROGRAM
4118
4119                          ;*****************************************************************
4120
4121                          .SBTTL  MAIN PROGRAM
4122
4123                          ;*****************************************************************
4124
4125  003120  023706  001220  MAIN:   CMP     @#STACK1,SP     ;CHECK STACK POINTER
4126  003124  001401                  BEQ     .+4             ;BR IF OK
```

```
4127  003126  000000                        HALT
4128  003130  005737  001222                TST     STATIN          ;***************
4129  003134  001404                        BEQ     .+12            ;TYPE ERROR RATE STATISTICS ?
4130  003136  005037  001222                CLR     STATIN          ;--)    BR IF NOT
4131  003142  004737  020362                JSR     PC,STATPR       ;  I    CLEAR THE INDICATOR
4132  003146  012703  000010                MOV     #8.,R3          ;  I    TYPE THE STATISTICS
4133  003152  012705  041560                MOV     #DUNIT,R5       ;(--    UNIT COUNTER
4134  003156  005715              1$:       TST     (R5)            ;ADDRESS OF 'DROP UNIT' TABLE
4135  003160  001005                        BNE     3$              ;SEE IF ENTRY AT PRESENT POSITION
4136  003162  062705  000002        2$:     ADD     #2,R5           ;BR IF THERE IS ONE
4137  003166  005303                        DEC     R3              ;INCREMENT TO NEXT TABLE POSITION
4138  003170  001372                        BNE     1$              ;DECREMENT UNIT COUNTER
4139  003172  000434                        BR      MAIN1           ;BR IF MORE TO CHECK
4140  003174  012701  041624        3$:     MOV     #AVAIL,R1       ;FINISHED HERE, GO TRY TO ASSIGN UNITS
4141  003200  005711              4$:       TST     (R1)            ;ADDRESS OF 'AVAILABLE' UNITS TABLE
4142  003202  001405                        BEQ     5$              ;SEE IF AT END OF TABLE
4143  003204  021115                        CMP     (R1),(R5)       ;BR IF AT END: GO CHECK 'WAIT' TABLE
4144  003206  001414                        BEQ     7$              ;IS UNIT IN 'AVAIL' THE ONE TO BE DROPPED
4145  003210  062701  000002                ADD     #2,R1           ;BR IF YES
4146  003214  000771                        BR      4$              ;INCREMENT 'AVAIL' TABLE ADDRESS
4147  003216  012701  041646        5$:     MOV     #WAIT,R1        ;CONTINUE LOOKING
4148  003222  005711              6$:       TST     (R1)            ;MOVE THE ADDRESS OF THE BUFFER WAIT TABLE
4149  003224  001756                        BEQ     2$              ;AT THE END OF THE 'WAIT' TABLE ?
4150  003226  021115                        CMP     (R1),(R5)       ;BR IF YES: SEE IF ANY MORE 'DROP' REQUESTS
4151  003230  001403                        BEQ     7$              ;UNIT IN THE 'WAIT' TABLE ?
4152  003232  062701  000002                ADD     #2,R1           ;BR IF IT IS
4153  003236  000771                        BR      6$              ;INCREMENT 'WAIT' TABLE ADDRESS
4154  003240  011100              7$:       MOV     (R1),R0         ;CONTINUE LOOK THROUGH THE 'WAIT' TABLE
4155  003242  104400  051725                TYPE    ,DEASSG         ;PUT THE UNIT'S BLOCK ADDRESS IN R0
4156  003246  004737  021110                JSR     PC,TYPEST       ;TYPE 'UNIT DEASSIGNED'
4157  003252  005015                        CLR     (R5)            ;TYPE THE UNIT'S ERROR RATE DATA
4158  003254  005011                        CLR     (R1)            ;CLEAR THE 'DROP UNIT' TABLE ENTRY
4159  003256  004737  015234                JSR     PC,CMPRES       ;REMOVE THE UNIT FROM THE 'AVAIL' OR 'WAIT' TABLE
4160  003262  000737                        BR      2$              ;COMPRESS THE RESPECTIVE TABLE
4161                                                                ;SEE IF ANY MORE UNITS
4162                                  ;LOOK FOR DRIVES TO BE ASSIGNED
4163
4164  003264  012703  000010        MAIN1:  MOV     #8.,R3          ;UNIT COUNT
4165  003270  005002                        CLR     R2              ;'AVAIL' INDEX
4166  003272  005004                        CLR     R4              ;ASSIGN LIST INDEX
4167  003274  005005                        CLR     R5              ;NEW UNIT INDEX
4168  003276  005765  041602        1$:     TST     NEWUNT(R5)      ;NEW UNIT IN THIS POSITION
4169  003302  001006                        BNE     3$              ;BR IF THERE IS
4170  003304  062705  000002        2$:     ADD     #2,R5           ;INCREMENT R5
4171  003310  005204                        INC     R4              ;INCREMENT ASSIGN INDEX
4172  003312  005303                        DEC     R3              ;DECREMENT UNIT COUNT
4173  003314  001370                        BNE     1$              ;BR IF MORE UNITS
4174  003316  000506                        BR      MAIN2           ;START OPERATIONS FOR THE AVAILABLE UNITS
4175  003320  142737  000007  001255  3$:     BICB    #7,UNIT+1       ;CLEAR PREVIOUS UNIT NUMBER
4176  003326  150437  001255                BISB    R4,UNIT+1       ;PRESENT UNIT NUMBER
4177  003332  104400  051277                TYPE    ,UNTMSG         ;'UNIT'
4178  003336  104400  001254                TYPE    ,UNIT           ;UNIT NUMBER
4179  003342  104400  051775                TYPE    ,ASGND          ;'ASSIGNED'
4180  003346  005762  041624        4$:     TST     AVAIL(R2)       ;AT END OF AVAILABLE TABLE
```

```
4181  003352  001403                          BEQ     5$              ;BR IF YES
4182  003354  062702  000002                  ADD     #2,R2           ;INCREMENT AVAILABLE TABLE INDEX
4183  003360  000772                          BR      4$              ;CONTINUE LOOKING FOR END OF TABLE
4184  003362  016562  041602  041624  5$:     MOV     NEWUNT(R5),AVAIL(R2)  ;MOVE ADDR OF UNIL INTO AVAIL LST
4185  003370  005065  041602                  CLR     NEWUNT(R5)      ;TAKE UNIT OUT OF NEW UNIT TABLE
4186  003374  112764  177777  041550          MOVB    #-1,ASNLST(R4)  ;SET UNIT ASSIGNED INDICATOR
4187  003402  016200  041624                  MOV     AVAIL(R2),R0    ;PUT STARTING ADDRESS OF BLOCK IN R0
4188  003406  113760  001330  000010          MOVB    MINSEC,$SEC(R0)  ;INITIAL SECTOR VALUE
4189  003414  113760  001324  000011          MOVB    MINTRK,$TRK(R0)  ;INITIAL TRACK VALUE
4190  003422  013760  001320  000012          MOV     MINCYL,$CYL(R0)  ;INITIAL CYLINDER VALUE
4191  003430  113760  001334  000030          MOVB    BEGCOD,$CODE(R0)  ;INITIAL COMMAND CODE
4192  003436  013701  001334                  MOV     BEGCOD,R1       ;GET THE ACTUAL OP CODE
4193  003442  116160  042054  000002          MOVB    COMTBL(R1),$COMND(R0)  ;OPERATION CODE
4194  003450  113760  001332  000034          MOVB    BEGPAT,$PATTC(R0)  ;PATTERN CODE
4195  003456  106360  000034                  ASLB    $PATTC(R0)      ;CONVERT CODE TO A TABLE INDEX
4196  003462  013760  001336  000020          MOV     BEGSIZ,$WRDL(R0)  ;BEGINNING RECORD SIZE
4197  003470  013760  001336  000004          MOV     BEGSIZ,$WRDM(R0)  ;VALUE FOR DATA TRANSFER
4198  003476  005460  000004                  NEG     $WRDM(R0)       ;MAKE IT INTO 2'S COMPLEMENT
4199  003502  012760  000400  000026          MOV     #256.,$SSEC(R0)  ;INITIAL VALUE OF SECTOR SIZE
4200  003510  132760  000001  000030          BITB    #1,$CODE(R0)    ;HEADER ORDER ?
4201  003516  001403                          BEQ     6$              ;BR IF NOT
4202  003520  062760  000004  000026          ADD     #4,$SSEC(R0)    ;ADD HEADER SIZE TO SECTOR SIZE
4203  003526  062702  000002          6$:     ADD     #2,R2           ;INCREMENT AVAILABLE TABLE POINTER
4204  003532  000664                          BR      2$              ;LOOK FOR MORE UNITS
4205
4206                                  ;GET PARAMETERS, BUFFER SPACE, AND START ORDERS FOR DRIVES IN
4207                                  ;       THE 'AVAILABLE' QUEUE
4208
4209  003534  005737  041646          MAIN2:  TST     WAIT            ;OUTSTANDING BUFFER REQUESTS
4210  003540  001110                          BNE     MAIN3           ;BR IF THERE ARE
4211  003542  005737  041624                  TST     AVAIL           ;ANY UNIT  WAITING FOR PARAMETERS
4212  003546  001545                          BEQ     IDLE            ;BRANCH IF ,ONE
4213  003550  013700  041624                  MOV     AVAIL,R0        ;CONTROL BLOCK ADDR IN R0
4214  003554  005760  000112                  TST     $NEXT(R0)       ;PARAMETERS BEEN SELECTED ?
4215  003560  001021                          BNE     2$              ;BR IF THEY HAVE
4216  003562  105760  000031                  TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR THE DRIVE ?
4217  003566  001403                          BEQ     1$              ;BR IF NOT
4218  003570  004737  015252                  JSR     PC,WRTPK        ;GET DATA PACK PARAMETERS
4219  003574  000415                          BR      3$              ;GET THE BUFFER
4220  003576  012701  041670          1$:     MOV     #PARQ,R1        ;ADDRESS OF THE PARAMETER QUEUE
4221  003602  020011          7$:     CMP     R0,(R1)         ;IS CURRENT UNIT IN THE QUEUE
4222  003604  001403                          BEQ     8$              ;BR IF IT IS
4223  003606  005721                          TST     (R1)+           ;AT END OF THE QUEUE
4224  003610  001403                          BEQ     9$              ;BR IF AT END
4225  003612  000773                          BR      7$              ;CONTINUE LOOKING
4226  003614  004737  015234          8$:     JSR     PC,CMPRES       ;COMPRESS THE TABLE
4227  003620  004737  014330          9$:     JSR     PC,SELPAR       ;SELECT THE PARAMETERS
4228  003624  004737  015036          2$:     JSR     PC,GETPAR       ;LOAD NEW PARAMETERS
4229  003630  005046          3$:     CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
4230  003632  004737  013416                  JSR     PC,GETBUF       ;GET BUFFER
4231  003636  012660  000006                  MOV     (SP)+,$BUF(R0)  ;MOVE BUFFER ADDR TO DPB
4232  003642  001426                          BEQ     5$              ;BR IF '0' ADDR (NO BUFFER)
4233  003644  004737  014004                  JSR     PC,FILBUF       ;FILL THE BUFFER
4234  003650  004737  014252                  JSR     PC,GODRIV       ;PUT CURRENT DPB IN DRIVER
```

```
4235  003654  012705  041526                MOV     #ORDERQ,R5      ;ADDRESS OF ORDER QUEUE IN R5
4236  003660  005725                        TST     (R5)+           ;END OF QUEUE ?
4237  003662  001376                        BNE     .-2             ;BR IF NOT
4238  003664  010045                        MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO QUEUE
4239  003666  105760  000031                TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR DRIVE ?
4240  003672  001005                        BNE     10$             ;BR IF EITHER
4241  003674  012705  041670                MOV     #PARQ,R5        ;PUT BLOCK INTO THE PARAMETER QUEUE
4242  003700  005725                4$:     TST     (R5)+           ;FIND THE END OF THE QUEUE
4243  003702  001376                        BNE     4$              ;BR IF NOT AT END OF QUEUE
4244  003704  010045                        MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO THE QUEUE
4245  003706  012701  041624        10$:    MOV     #AVAIL,R1       ;R1 CONTAINS THE TABLE ADDRESS
4246  003712  004737  015234                JSR     PC,CMPRES       ;COMPRESS THE AVAILABILITY TABLE
4247  003716  000706                        BR      MAIN2           ;CONTINUE PROCESSING
4248  003720  005005                5$:     CLR     R5              ;R5 USED AS 'WAIT' POINTER
4249  003722  000240                        NOP                     ;DEBUGGING AID
4250  003724  005765  041646                TST     WAIT(R5)        ;AT END OF TABLE
4251  003730  001403                        BEQ     6$              ;BR IF YES
4252  003732  062705  000002                ADD     #2,R5           ;INCREMENT R5
4253  003736  000770                        BR      5$              ;CONTINUE SEARCHING
4254  003740  013765  041624  041646  6$:    MOV     AVAIL,WAIT(R5)  ;MOVE ENTRY FROM AVAILABLE TO WAIT
4255  003746  012701  041624                MOV     #AVAIL,R1       ;TABLE TO COMPRESS
4256  003752  004737  015234                JSR     PC,CMPRES       ;COMPRESS THE INDICATED TABLE
4257  003756  000137  004062                JMP     IDLE            ;WAIT FOR A UNIT TO BECOME READY
4258
4259                                ;GET BUFFER ASSIGNMENTS FOR DRIVES INT THE 'BUFFER WAIT' QUEUE
4260
4261  003762  013700  041646        MAIN3:  MOV     WAIT,R0         ;MOVE THE 'WAIT' ENTRY TO R0
4262  003766  005046                        CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
4263  003770  004737  013416                JSR     PC,GETBUF       ;TRY TO GET A BUFFER
4264  003774  012660  000006                MOV     (SP)+,$BUF(R0)  ;MOVE THE BUFFER ADDR TO THE DPB
4265  004000  001002                        BNE     1$              ;BR IF P BUFFER WAS ASSIGNED
4266  004002  000137  004062                JMP     IDLE            ;NO BUFFER AVAILABLE YET
4267  004006  004737  014004        1$:     JSR     PC,FILBUF       ;FILL THE BUFFER
4268  004012  004737  014252                JSR     PC,GODRIV       ;PUT THE ENTRY IN THE DRIVER
4269  004016  012705  041526                MOV     #ORDERQ,R5      ;ADDRESS OF ORDER QUEUE IN R5
4270  004022  005725                        TST     (R5)+           ;AT END OF THE QUEUE
4271  004024  001376                        BNE     .-2             ;BR IF NOT
4272  004026  010045                        MOV     R0,-(R5)        ;PUT BLOCK ADDRESS IN QUEUE
4273  004030  105760  000031                TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR DRIVE ?
4274  004034  001005                        BNE     3$              ;BR IF YES, DON'T PUT BLOCK INTO 'PARQ'
4275  004036  012705  041670                MOV     #PARQ,R5        ;FIND THE END OF THE PARAMETER QUEUE
4276  004042  005725                2$:     TST     (R5)+           ;OPEN SLOT IN THE QUEUE ?
4277  004044  001376                        BNE     2$              ;BR IF NOT
4278  004046  010045                        MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO THE QUEUE
4279  004050  012701  041646        3$:     MOV     #WAIT,R1        ;ADDRESS OF TABLE TO TO COMPRESS
4280  004054  004737  015234                JSR     PC,CMPRES       ;COMPRESS THE WAIT TABLE
4281  004060  000625                        BR      MAIN2           ;LOOK FOR MORE ENTRIES
4282
4283                                ;WAIT FOR AN ORDER TO FINISH
4284
4285  004062  023706  001220        IDLE:   CMP     @#STACK1,SP     ;CHECK STACK POINTER
4286  004066  001401                        BEQ     .+4             ;BR IF OK
4287  004070  000000                        HALT                    ;STACK POINTER NOT 1100 **************
4288  004072  012701  041526                MOV     #ORDERQ,R1      ;ADDRESS OF THE ORDER QUEUE IN R1
```

```
4289  004076  012100                  1$:     MOV     (R1)+,R0        ;PUT BLOCK ADDRESS INTO R0
4290  004100  001433                          BEQ     IDLE1           ;BR IF END OF QUEUE
4291  004102  005760  000016                  TST     $STATUS(R0)     ;SEE IF UNIT FINISHED
4292  004106  001773                          BEQ     1$              ;BR IF UNIT NOT FINISHED
4293  004110  162701  000002                  SUB     #2,R1           ;CORRECT THE QUEUE POINTER
4294  004114  010146                          MOV     R1,-(SP)        ;SAVE THE QUEUE ADDRESS
4295  004116  004737  013262                  JSR     PC,STATIS       ;ACCUMULATE STATISTICS FOR UNIT IN R0
4296  004122  000240                          NOP                     ;DEBUGGING AID
4297  004124  004737  004372                  JSR     PC,PROCES       ;PROCESS END OF ORDER
4298  004130  005037  016330                  CLR     PRT2B           ;CLEAR THE BAD TRK/SEC ERROR INDICATOR
4299  004134  004737  023616                  JSR     PC,ABNRML       ;SEE IF ANY UNITS HAVE TOO MANY ERRORS
4300  004140  004737  023640                  JSR     PC,NRML         ;SEE IF ANY UNIT HAS XFERED 3X10↑9 BITS
4301  004144  012601                          MOV     (SP)+,R1        ;RESTORE THE ORDER TABLE INDEX
4302  004146  012705  041624                  MOV     #AVAIL,R5       ;FIND THE END OF THE 'AVAILABLE' TABLE
4303  004152  005725                  2$:     TST     (R5)+           ;END OF THE TABLE ?
4304  004154  001376                          BNE     2$              ;BR IF NOT AT END OF LIST
4305  004156  011145                          MOV     (R1), -(R5)     ;MOV THE BLOCK ADDRESS INTO THE TABLE
4306  004160  004737  015234                  JSR     PC,CMPRES       ;COMPRESS THE ORDER QUEUE
4307  004164  004737  013556                  JSR     PC,RELBUF       ;RESTORE BUFFER
4308  004170  005737  041670          IDLE1:  TST     PARQ            ;ENTRY IN THE PARAMETER QUEUE ?
4309  004174  001410                          BEQ     1$              ;BR IF NOT
4310  004176  013700  041670                  MOV     PARQ,R0         ;PUT THE BLOCK ADDRESS INTO R0
4311  004202  004737  014330                  JSR     PC,SELPAR       ;GET THE PARAMETERS FOR NEXT OPERATION
4312  004206  012701  041670                  MOV     #PARQ,R1        ;SETUP TO COMPRESS THE TABLE
4313  004212  004737  015234                  JSR     PC,CMPRES       ;COMPRESS THE PARAMETER QUEUE
4314  004216  000137  003120          1$:     JMP     MAIN            ;CONTINUE THE LOOP
4315
4316                                          ;SETUP TO REFORMAT AN ERROR SECTOR
4317
4318  004222  032737  000001  177570  REFMT:  BIT     #SW0,SWR        ;READ ONLY SWITCH SET ?
4319  004230  001057                          BNE     REFMTX          ;BR IF IT IS
4320  004232  032737  000200  177570          BIT     #SW7,SWR        ;SWITCH 7 SET ?
4321  004240  001053                          BNE     REFMTX          ;BR IF IT IS
4322  004242  005737  001302                  TST     FORMAT          ;WRITE HEADER & DATA ORDERS ALLOWED ?
4323  004246  001450                          BEQ     REFMTX          ;BR IF NOT
4324  004250  016060  000222  000106          MOV     $RHCC(R0),$NCYL(R0)  ;USE PRESENT CYLINDER
4325  004256  005046                          CLR     -(SP)           ;CLEAR STACK
4326  004260  005046                          CLR     -(SP)           ;FOR SECTOR & TRACK
4327  004262  004737  020036                  JSR     PC,READDR       ;GET CORRECTED SECTOR-TRACK ADDRESSES
4328  004266  112660  000105                  MOVB    (SP)+,$NTRK(R0) ;TRACK ADDR TO DPB
4329  004272  112660  000104                  MOVB    (SP)+,$NSEC(R0) ;SECTOR ADDR TO DPB
4330  004276  012760  000404  000110          MOV     #260.,$NWRDL(R0)  ;WORD COUNT FOR FORMAT
4331  004304  023727  001272  000404          CMP     MAXDL,#260.     ;CAN A FULL SECTOR BE WRITTEN ?
4332  004312  103003                          BHIS    1$              ;BR IF IT CAN
4333  004314  013760  001272  000110          MOV     MAXDL,$NWRDL(R0)  ;PUT TRANSFER SIZE INTO THE DPB
4334  004322  112760  000003  000102  1$:     MOVB    #3,$NCODE(R0)   ;COMMAND CODE
4335  004330  004737  015010                  JSR     PC,GETPAT       ;GET A PATTERN
4336  004334  110560  000103                  MOVB    R5,$NPATC(R0)   ;PATTERN CODE TO CONTROL BLOCK
4337  004340  012760  177777  000112          MOV     #-1,$NEXT(R0)   ;SET PARAMETERS SELECTED INDICATOR
4338  004346  012701  041670                  MOV     #PARQ,R1        ;SET UP TO SEE IF BLOCK IN THE PARAMETER QUEUE
4339  004352  005711                  2$:     TST     (R1)            ;SEE IF AT END OF TABLE
4340  004354  001405                          BEQ     REFMTX          ;BR IF AT END
4341  004356  020021                          CMP     R0,(R1)+        ;SEE IF BLOCK AT PRESENT POSITION
4342  004360  001374                          BNE     2$              ;BR IF NOT
```

```
4343  004362  005041                          CLR     -(R1)           ;CLEAR THE ENTRY
4344  004364  004737  015234                  JSR     PC,CMPRES       ;COMPRESS THE TABLE
4345  004370  000207                  REFMTX: RTS     PC              ;RETURN
4346
4347                                  ;PROCESS THE ORDER TERMINATION
4348
4349  004372  005760  000016          PROCES: TST     $TATUS(R0)      ;SEE IF DRIVER SIGNALED AN ERROR
4350  004376  100427                          BMI     ERPROC          ;BR IF ERROR
4351  004400  032760  100000  000164          BIT     #BIT15,$RHCS1(R0)  ;SEE IF 'SC' SET
4352  004406  001410                          BEQ     1$              ;BR IF NOT SET
4353  004410  032760  040000  000164          BIT     #BIT14,$RHCS1(R0)  ;SEE IF 'TRE' SET
4354  004416  001017                          BNE     ERPROC          ;BR IF SET
4355  004420  032760  040000  000176          BIT     #BIT14,$RHDS1(R0)  ;SEE IF 'ERR' SET
4356  004426  001013                          BNE     ERPROC          ;BR IF SET
4357  004430  004737  010406          1$:     JSR     PC,CKERR        ;NO ERROR, CHECK ERROR BITS ANYWAY
4358  004434  004737  010506                  JSR     PC,CKBUS        ;NO ERROR, CHECK BUS ADDR & WC
4359  004440  032737  000004  177570          BIT     #SW02,SWR       ;NORMAL DATA COMPARE ?
4360  004446  001002                          BNE     2$              ;BR IF NOT
4361  004450  004737  010572                  JSR     PC,CMPAR        ;NO ERROR, COMPARE DATA
4362  004454  000207                  2$:     RTS     PC              ;RETURN
4363
4364                                  ;ORDER TERMINATED WITH AN ERROR - PROCESS THE ERROR
4365
4366  004456  032760  000200  000016  ERPROC: BIT     #BIT07,$TATUS(R0) ;DONE BIT SET ?
4367  004464  001402                          BEQ     ERPRC1          ;BR IF ORDER DIDN'T COMPLETE NORMALLY
4368  004466  000137  005114                  JMP     DONE            ;PROCESS ERROR WITH 'DONE' BIT SET
4369
4370                                  ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE NOT' BITS
4371
4372  004472  142737  000007  001255  ERPRC1: BICB    #7,UNIT+1       ;CLEAR OLD UNIT NUMBER
4373  004500  151037  001255                  BISB    (R0),UNIT+1     ;SET PRESENT UNIT NUMBER
4374  004504  032760  010000  000016          BIT     #BIT12,$TATUS(R0) ;SEE IF DRIVE WAS UNSAFE
4375  004512  001025                          BNE     PUNSAF          ;BR IF YES
4376  004514  032760  004000  000016          BIT     #BIT11,$TATUS(R0) ;PARITY ERROR OCCURRED
4377  004522  001053                          BNE     UCPAR           ;BR IF IT DID
4378  004524  032760  002000  000016          BIT     #BIT10,$TATUS(R0) ;FATAL PARITY ERROR?
4379  004532  001054                          BNE     FALPAR          ;BR IF THERE IS ONE
4380  004534  032760  001000  000016          BIT     #BIT09,$TATUS(R0) ;TIMEOUT?
4381  004542  001072                          BNE     SWTIM           ;BR IF YES
4382  004544  032760  040000  000016          BIT     #BIT14,$TATUS(R0) ;UNIT WENT OFFLINE ?
4383  004552  001107                          BNE     OFLIN           ;BR IF IT DID
4384  004554  032760  000004  000016          BIT     #BIT2,$TATUS(R0) ;PORT REQUEST TIME OUT ?
4385  004562  001135                          BNE     PRTIM           ;BR IF IT DID
4386  004564  000207                          RTS     PC              ;ERROR. RETURN
4387
4388                                  ;DRIVE IS PERSISTENTLY UNSAFE
4389
4390  004566  104400  001157          PUNSAF: TYPE    ,$CRLF          ;CR-LF
4391  004572  104400  044210                  TYPE    ,EM12           ;'DRIVE UNSAFE' MESSAGE
4392  004576  104400  051750                  TYPE    ,DRNUM          ;DRIVE NUMBER
4393  004602  104400  001254                  TYPE    ,UNIT           ;TYPE THE DRIVE NUMBER
4394  004606  104400  001157                  TYPE    ,$CRLF          ;CR-LF
4395  004612  004737  015714                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4396  004616  104422  044210                  DISPLY  ,EM12           ;PERSISTENT DEVICE UNSAFE MESSAGE
```

```
4397  004622  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4398  004626  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4399  004632  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF THE ERROR MESSAGE
4400  004636  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4401  004642  004737  017534                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4402  004646  000137  023534                  JMP     DROP            ;DROP THE UNIT
4403
4404                                  ;UNCORRECTABLE MASSBUS PARITY ERROR OCCURED
4405
4406  004652  104400  001157          UCPAR:  TYPE    ,$CRLF          ;CR-LF
4407  004656  104400  044112                  TYPE    ,EM10           ;'UNCORRECTABLE PARITY ERROR' MESSAGE
4408  004662  000404                  BR      FALPR1          ;FINISH PROCESSING THE ERROR
4409
4410                                  ;'FATAL' MASSBUS PARITY ERROR OCCURED
4411
4412  004664  104400  001157          FALPAR: TYPE    ,$CRLF          ;CR-LF
4413  004670  104400  044155                  TYPE    ,EM11           ;'FATAL PARITY ERROR' MESSAGE
4414  004674  104400  051750          FALPR1: TYPE    ,DRNUM          ;DRIVE NUMBER
4415  004700  104400  001254                  TYPE    ,UNIT           ;TYPE THE DRIVE NUMBER
4416  004704  104400  001157                  TYPE    ,$CRLF          ;CR-LF
4417  004710  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4418  004714  032737  100000  177570          BIT     #SW15,SWR       ;HALT ON ERROR ?
4419  004722  001401                  BEQ     .+4             ;BR IF NOT
4420  004724  000000                  HALT                    ;ERROR HALT
4421  004726  000207                  RTS     PC
4422
4423                                  ;SOFTWARE TIMEOUT OCCURED
4424
4425  004730  004737  015714          SWTIM:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4426  004734  104422  044241                  DISPLY  ,EM13           ;PRINT THE TIME OUT MESSAGE
4427  004740  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4428  004744  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4429  004750  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
4430  004754  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4431  004760  004737  017534                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4432  004764  000240                  NOP
4433  004766  000240                  NOP
4434  004770  000207                  RTS     PC              ;RETURN
4435
4436                                  ;DRIVE WENT OFFLINE
4437
4438  004772  104400  001157          OFLIN:  TYPE    ,$CRLF          ;CR-LF
4439  004776  104400  044313                  TYPE    ,EM14           ;'DRIVE WENT OFFLINE' MESSAGE
4440  005002  104400  051750                  TYPE    ,DRNUM          ;DRIVE NUMBER
4441  005006  104400  001254                  TYPE    ,UNIT           ;TYPE THE DRIVE NUMBER
4442  005012  104400  001157                  TYPE    ,$CRLF          ;CR-LF
4443  005016  004737  015714                  JSR     PC,LINE1        ;PRINT LINE 1 OF THE ERROR MESSAGE
4444  005022  104422  044313                  DISPLY  ,EM14           ;PRINT OFFLINE MESSAGE
4445  005026  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF THE ERROR MESSAGE
4446  005032  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF THE ERROR MESSAGE
4447  005036  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF THE ERROR MESSAGE
4448  005042  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4449  005046  004737  017534                  JSR     PC,LINE7        ;PRINT LINE 7 OF THE ERROR MESSAGE
4450  005052  000137  023534                  JMP     DROP            ;DROP THE UNIT
```

```
4451
4452                                    ;PORT REQUEST TIMEOUT ERROR
4453
4454  005056  004737  015714    PRTIM:  JSR     PC,LINE1        ;TYPE LINE 1 OF THE ERROR MESSAGE
4455  005062  104422  044335            DISPLY  ,EM15           ;PRINT PORT TIME OUT MESSAGE
4458  005066  004737  015760            JSR     PC,LINE2        ;TYPE LINE 2 OF THE ERROR MESSAGE
 (1)  005072  004737  016332            JSR     PC,LINE3        ;TYPE LINE 3 OF THE ERROR MESSAGE
 (1)  005076  004737  017070            JSR     PC,LINE4        ;TYPE LINE 4 OF THE ERROR MESSAGE
4459  005102  004737  021270            JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4460  005106  004737  017534            JSR     PC,LINE7        ;TYPE LINE 7 OF THE ERROR MESSAGE
4461  005112  000207                    RTS     PC              ;RETURN
4462
4463                                    ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE' BITS SET
4464
4465  005114  032760  000030  000016  DONE:   BIT     #BIT04!BIT03,$TATUS(R0) ;UNSAFE OCCURRED
4466  005122  001402                    BEQ     .+6             ;BR IF NOT
4467  005124  000137  010142            JMP     UNSAF           ;REPORT UNSAFE
4468  005130  032760  040000  000174    BIT     #BIT14,$RHCS2(R0)  ;IS 'WCE' SET ?
4469  005136  001402                    BEQ     .+6             ;BR IF NOT
4470  005140  000137  006060            JMP     WCKER           ;REPORT 'WCE'
4471  005144  032760  040000  000176    BIT     #BIT14,$RHDS1(R0) ;CHECK 'ERR'
4472  005152  001002                    BNE     .+6             ;BR IF SET
4473  005154  000137  007706            JMP     TRFER           ;PROCESS 'TRE'
4474  005160  032760  000020  000200    BIT     #BIT04,$RHER1(R0) ;'FMT' SET?
4475  005166  001402                    BEQ     .+6             ;BR IF NOT SET
4476  005170  000137  006536            JMP     CKFMT           ;CHECK FORMAT ERROR
4477  005174  032760  000200  000200    BIT     #BIT07,$RHER1(R0) ;'HCE' SET?
4478  005202  001402                    BEQ     .+6             ;BR IF NOT SET
4479  005204  000137  006736            JMP     CKHCE           ;CHECK 'HCE' ERROR
4480  005210  032760  000400  000200    BIT     #BIT08,$RHER1(R0) ;'HCRC' SET?
4481  005216  001402                    BEQ     .+6             ;BR IF NOT
4482  005220  000137  006354            JMP     HCRCER          ;PROCESS 'HCRC'
4483  005224  032760  020000  000200    BIT     #BIT13,$RHER1(R0) ;'OPI' SET?
4484  005232  001402                    BEQ     .+6             ;BR IF NOT SET
4485  005234  000137  007242            JMP     OPIER           ;REPORT 'OPI'
4486  005240  032760  000010  000200    BIT     #BIT3,$RHER1(R0) ;'PAR' SET?
4487  005246  001402                    BEQ     .+6             ;BR IF NOT SET
4488  005250  000137  007374            JMP     PARER           ;REPORT 'PAR'
4489  005254  032760  000040  000200    BIT     #BIT5,$RHER1(R0) ;'WCF' SET?
4490  005262  001402                    BEQ     .+6             ;BR IF NOT SET
4491  005264  000137  010044            JMP     WCFER           ;REPORT 'WCF'
4492  005270  032760  002000  000200    BIT     #BIT10,$RHER1(R0) ;'IAE' SET?
4493  005276  001402                    BEQ     .+6             ;BR IF NOT SET
4494  005300  000137  007466            JMP     IAEER           ;REPORT 'IAE'
4495  005304  032760  004000  000200    BIT     #BIT11,$RHER1(R0) ;'WLE' SET?
4496  005312  001402                    BEQ     .+6             ;BR IF NOT SET
4497  005314  000137  007520            JMP     WLEER           ;REPORT 'WLE'
4498  005320  032760  001000  000200    BIT     #BIT9,$RHER1(R0) ;'AOE' SET?
4499  005326  001405                    BEQ     1$              ;BR IF NOT SET
4500  005330  032760  002000  000176    BIT     #BIT10,$RHDS1(R0) ;'LST' SET?
4501  005336  001401                    BEQ     1$              ;BR IF NOT SET
4502  005340  000207                    RTS     PC              ;'AOE' & 'LST' SET, EXIT
4503  005342  032760  010000  000200  1$:     BIT     #BIT12,$RHER1(R0)  ;SEE IF 'DTE' SET
4504  005350  001402                    BEQ     .+6             ;BR IF NOT
```

```
4505  005352  000137  007352                  JMP     DTEER           ;REPORT 'DTE' ERROR
4506  005356  005760  000200                  TST     $RHER1(R0)      ;SEE IF 'DCK' SET
4507  005362  100002                          BPL     .+6             ;BR IF NOT
4508  005364  000137  005410                  JMP     DCKER           ;PROCESS 'DCK'
4509  005370  032760  140000  000226          BIT     #BIT15!BIT14,$RHER3(R0) ;'SKI' OR 'OCYL' SET
4510  005376  001402                          BEQ     .+6             ;BR IF NOT SET
4511  005400  000137  010006                  JMP     SKIER           ;REPORT ERROR
4512  005404  000137  006504                  JMP     DRVER           ;REPORT DRIVE ERROR
4513
4514                                  ;PROCESS DATA ('DCK') CHECK ERROR
4515
4516  005410  004737  015560          DCKER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT A BAD SPOT ON THE PACK
4517  005414  000207                          RTS     PC              ;IT IS, DON'T REPORT IT
4518  005416  004737  015714                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4519  005422  104422  044412                  DISPLY  .EM21           ;DATA CHECK ERROR
4520  005426  004737  015760          DCKER1: JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4521  005432  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4522  005436  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
4523  005442  004737  012522                  JSR     PC,PRTBAD       ;SEE IF BAD SECTOR TO BE PRINTED
4524  005446  012760  110100  000024          MOV     #BIT15!BIT12!BIT06,$MASK(R0)  ;LOAD ERROR MASK
4525  005454  032760  010100  000200          BIT     #BIT12!BIT06,$RHER1(R0)  ;CHECK 'DTE' & 'ECH'
4526  005462  001003                          BNE     1$              ;BR IF SET
4527  005464  004737  017424                  JSR     PC,LINE6        ;PRINT LINE 6 OF ERROR MESSAGE
4528  005470  000530                          BR      8$              ;FINISH THE ERROR REPORT
4529  005472  012760  010000  000022  1$:     MOV     #10000,$RETRY(R0)  ;RETRY COUNT
4530  005500  005001                          CLR     R1              ;R1 IS OFFSET CODE POINTER
4531  005502  004737  013556                  JSR     PC,RELBUF       ;RELEASE THE BUFFER
4532  005506  005046                          CLR     -(SP)           ;MAKE ROOM FOR THE SECTOR ADDRESS
4533  005510  005046                          CLR     -(SP)           ;MAKE ROOM FOR THE TRACK ADDRESS
4534  005512  004737  020036                  JSR     PC,READDR       ;GET THE ADDRESS OF THE ERROR SECTOR
4535  005516  112660  000011                  MOVB    (SP)+,$TRK(R0)  ;TRACK ADDRESS OF ERROR SECTOR
4536  005522  112660  000010                  MOVB    (SP)+,$SEC(R0)  ;SECTOR ADDRESS OF ERROR SECTOR
4537  005526  016060  000222  000012          MOV     $RHCC(R0),$CYL(R0)  ;PRESENT CYLINDER
4538  005534  026060  000026  000020          CMP     $SSEC(R0),$WRDL(R0)  ;SEE IF TRANSFER LENGTH LESS THAN 1 SECTOR
4539  005542  103010                          BHIS    15$             ;BR IF IT IS; USE PRESENT TRANSFER LENGTH
4540  005544  016060  000026  000020          MOV     $SSEC(R0),$WRDL(R0)  ;CHANGE TRANSFER SIZE TO 1 SECTOR
4541  005552  016060  000020  000004          MOV     $WRDL(R0),$WRDM(R0)  ;SETUP WORD COUNT FOR OPERATION
4542  005560  005460  000004                  NEG     $WRDM(R0)       ;CHANGE COUNT TO 2'S COMP
4543  005564  005046                  15$:    CLR     -(SP)           ;SPACE FOR NEW BUFFER ADDRESS
4544  005566  004737  013416                  JSR     PC,GETBUF       ;GET A BUFFER
4545  005572  012660  000006                  MOV     (SP)+,$BUF(R0)  ;NEW BUFFER ADDRESS TO DPB
4546  005576  004737  014252          2$:     JSR     PC,GODRIV       ;RETRY
4547  005602  005760  000016          3$:     TST     $STATUS(R0)     ;TEST FOR DONE
4548  005606  001775                          BEQ     3$              ;BR IF NOT DONE
4549  005610  100070                          BPL     10$             ;BR IF NOT ERROR
4550  005612  032760  000200  000016          BIT     #BIT7,$STATUS(R0)  ;SEE IF ORDER TERMINIATED NORMALLY
4551  005620  001511                          BEQ     14$             ;BR IF NOT
4552  005622  036060  000024  000200          BIT     $MASK(R0),$RHER1(R0)  ;LOOK AT CURRENT ERROR
4553  005630  001427                          BEQ     5$              ;BR IF DIFFERENT ERROR
4554  005632  032760  010100  000200          BIT     #BIT12!BIT6,$RHER1(R0)  ;'ECH' OR 'DTE' STILL SET ?
4555  005640  001440                          BEQ     7$              ;BR IF NEITHER SET
4556  005642  105260  000022                  INCB    $RETRY(R0)      ;INCREMENT RETRY COUNT
4557  005646  105360  000023                  DECB    $RETRY+1(R0)    ;DECREMENT RETRY LIMIT
4558  005652  001351                          BNE     2$              ;BR IF MORE RETRY TO GO
```

```
 4559  005654  005201                          INC     R1                      ;INCREMENT TABLE INDEX
 4560  005656  116137  042142  041437          MOVB    OFFCOD(R1),GENDPB+$FMT  ;OFFSET CODE
 4561  005664  001437                          BEQ     9$                      ;BR IF END OF OFFSET TABLE
 4562  005666  112760  000002  000023          MOVB    #2,$RETRY+1(R0)         ;NEW RETRY LIMIT
 4563  005674  004737  012752                  JSR     PC,OFFST                ;OFFSET
 4564  005700  005737  041454          4$:     TST     GENDPB+$STATUS          ;SEE IF FINISHED WITH OFFSET
 4565  005704  001775                          BEQ     4$                      ;BR IF NOT
 4566  005706  100333                          BPL     2$                      ;BR IF OFFSET OK
 4567  005710  004737  020002          5$:     JSR     PC,LINE8                ;PRINT LINE 8 OF ERROR MESSAGE
 4568  005714  004737  021174          6$:     JSR     PC,INCHRD               ;INCREMENT 'HARD' ERROR COUNT
 4569  005720  004737  021270                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
 4570  005724  004737  017534                  JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
 4571  005730  004737  012522                  JSR     PC,PRTBAD               ;PRINT THE BAD SECTOR
 4572  005734  004737  004222                  JSR     PC,REFMT                ;REFORMAT THE ERROR SECTOR
 4573  005740  000207                          RTS     PC                      ;RETURN
 4574  005742  004737  017464          7$:     JSR     PC,LINE6B               ;PRINT LINE 6B OF ERROR MESSAGE
 4575  005746  004737  017362                  JSR     PC,LINE5B               ;PRINT LINE 5B OF THE ERROR MESSAGE
 4576  005752  004737  021150          8$:     JSR     PC,INCSOF               ;INCREMENT 'SOFT' ERROR COUNT
 4577  005756  004737  011736                  JSR     PC,ECC                  ;CORRECT THE ERROR USING ECC AND CHECK IT
 4578  005762  000407                          BR      11$                     ;COMPARE THE BUFFER
 4579  005764  004737  017524          9$:     JSR     PC,LINE6D               ;PRINT LINE 6D OF ERROR MESSAGE
 4580  005770  000751                          BR      6$                      ;INCREMENT ERROR COUNT
 4581  005772  004737  017436          10$:    JSR     PC,LINE6A               ;PRINT LINE 6A OF ERROR MESSAGE
 4582  005776  004737  021150                  JSR     PC,INCSOF               ;INCREMENT 'SOFT' ERROR COUNT
 4583  006002  112737  000001  011321  11$:     MOVB    #1,FRSTER               ;SET 'DCKER' INDICATOR
 4584  006010  004737  010622                  JSR     PC,CMPARD               ;COMPARE THE BUFFER
 4585  006014  005737  011326                  TST     ERCTR                   ;SEE IF COMPARE ERROR DETECTED
 4586  006020  001006                          BNE     13$                     ;BR IF ONE DID
 4587  006022  004737  021270                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
 4588  006026  104422  050661                  DISPLY  ,LIN9G                  ;'DATA COMPARE OK' MESSAGE
 4589  006032  004737  017534          12$:    JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
 4590  006036  004737  004222          13$:    JSR     PC,REFMT                ;REFORMAT THE ERROR SECTOR
 4591  006042  000207                          RTS     PC                      ;RETURN
 4592  006044  004737  021270          14$:    JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
 4593  006050  104422  050447                  DISPLY  ,LIN8M                  ;'DIFFERENT ERROR DURING RETRY'
 4594  006054  000137  004472                  JMP     ERPRC1                  ;SEE WHICH ERROR
 4595
 4596                                  ;WRITE CHECK ERROR PROCESSING
 4597
 4598  006060  004737  015714          WCKER:  JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
 4599  006064  032760  100000  000200          BIT     #BIT15,$RHER1(R0)       ;SEE IF 'DCK' SET ALSO
 4600  006072  001032                          BNE     2$                      ;BR IF IT IS
 4601  006074  104422  044516                  DISPLY  ,EM23                   ;PRINT WCE & DCK NOT
 4602  006100  005060  000024                  CLR     $MASK(R0)               ;CLEAR ERROR MASK
 4605  006104  004737  015760                  JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
  (1)  006110  004737  016332                  JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
  (1)  006114  004737  017070                  JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
  (1)  006120  004737  017170                  JSR     PC,LINE5                ;PRINT LINE 5 OF ERROR MESSAGE
 4606  006124  004737  021270                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
 4607  006130  012760  001400  000022          MOV     #1400,$RETRY(R0)        ;RETRY LIMIT
 4608  006136  004737  013060                  JSR     PC,RETRY                ;RETRY THE OPERATION
 4609  006142  000403                          BR      1$                      ;RETRY UNSUCESSFUL
 4610  006144  004737  017474                  JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
 4611  006150  000465                          BR      8$                      ;FINISH PROCESSING THE ERROR
```

```
 4612  006152  004737  017524          1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
 4613  006156  000471                          BR      10$             ;FINISH PROCESSING THE ERROR
 4614  006160  004737  015560          2$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT ON THE PACK
 4615  006164  000207                          RTS     PC              ;BR IF IT IS
 4616  006166  104422  044443                  DISPLY  .EM22           ;WCK & DCK ERRS
 4619  006172  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
  (1)  006176  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
  (1)  006202  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
  (1)  006206  004737  017170                  JSR     PC,LINE5        ;PRINT LINE 5 OF ERROR MESSAGE
 4620  006212  032760  000100  000200          BIT     #BIT06,$RHER1(R0)  ;ECH SET ALSO ?
 4621  006220  001441                          BEQ     8$              ;FINISH PROCESSING THE ERROR
 4622  006222  012760  010000  000022  3$:     MOV     #10000,$RETRY(R0)  ;RETRY LIMIT - 16 (10)
 4623  006230  004737  014252          4$:     JSR     PC,GOORIV       ;RETRY THE ORDER
 4624  006234  005760  000016          5$:     TST     $TATUS(R0)      ;ORDER FINISHED ?
 4625  006240  001775                          BEQ     5$              ;BR IF NOT
 4626  006242  100405                          BMI     6$              ;BR IF ERROR ON ORDER
 4627  006244  105260  000022                  INCB    $RETRY(R0)      ;INCREMENT RETRY COUNT
 4628  006250  004737  017474                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
 4629  006254  000430                          BR      9$              ;FINISH ERROR PROCESSING
 4630  006256  105260  000022          6$:     INCB    $RETRY(R0)      ;INCREMENT RETRY COUNT
 4631  006262  105360  000023                  DECB    $RETRY+1(R0)    ;DECREMENT RETRY LIMIT
 4632  006266  001731                          BEQ     1$              ;BR IF AT RETRY LIMIT
 4633  006270  032760  100000  000200          BIT     #BIT15,$RHER1(R0)  ;'DCK' SET
 4634  006276  001407                          BEQ     7$              ;BR IF NOT - DIFFERENT ERROR
 4635  006300  032760  000100  000200          BIT     #BIT06,$RHER1(R0)  ;'ECH' ALSO SET ?
 4636  006306  001350                          BNE     4$              ;BR IF IT IS; RETRY ORDER
 4637  006310  004737  017474                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
 4638  006314  000403                          BR      8$              ;FINISH PROCESSING ERROR
 4639  006316  004737  020002          7$:     JSR     PC,LINE8        ;PRINT LINE 8 - 'DIFFERENT ERROR '
 4640  006322  000405                          BR      9$              ;FINISH PROCESSING ERROR
 4641  006324  004737  021270          8$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERRCR COUNT
 4642  006330  004737  017534                  JSR     PC,LINE7        ;FINISH THE ERROR MESSAGE
 4643  006334  000207                          RTS     PC              ;RETURN
 4644  006336  004737  021270          9$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 4645  006342  004737  017534          10$:    JSR     PC,LINE7        ;FINISH THE ERROR MESSAGE
 4646  006346  004737  004222                  JSR     PC,REFMT        ;REFORMAT THE SECTOR IN ERROR
 4647  006352  000207                          RTS     PC              ;RETURN
 4648
 4649                                  ;REPORT 'HCRC' ERROR
 4650
 4651  006354  004737  015560          HCRCER: JSR     PC,SPOTCK       ;SEE IF ERROR AT PACK BAD SPOT
 4652  006360  000207                          RTS     PC              ;RETURN IF IT IS
 4653  006362  004737  015714                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 4654  006366  104422  044371                  DISPLY  .EM20           ;REPORT 'HCRC'
 4655  006372  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
 4656  006376  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
 4657  006402  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
 4658  006406  032760  040000  000174          BIT     #BIT14,$RHCS2(R0)  ;'WCE' ERROR ALSO ?
 4659  006414  001402                          BEQ     .+6             ;BR IF NOT
 4660  006416  004737  017170                  JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
 4661  006422  004737  021150                  JSR     PC,INCSOF       ;INCREMENT 'SOFT' ERROR COUNT
 4662  006426  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 4663  006432  012760  000400  000024          MOV     #BIT8,$MASK(R0) ;SET ERROR MASK
 4664  006440  012760  001400  000022          MOV     #1400,$RETRY(R0)  ;RETRY LIMIT
```

```
4665  006446  004737  013060              JSR     PC,RETRY        ;RETRY ORDER
4666  006452  000405                      BR      1$              ;RETRY NOT SUCESSFUL
4667  006454  004737  017474              JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
4668  006460  004737  017534              JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4669  006464  000207                      RTS     PC              ;RETURN
4670  006466  004737  017524      1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
4671  006472  004737  017534              JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4672  006476  004737  004222              JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
4673  006502  000207                      RTS     PC              ;RETURN
4674
4675                                  ;REPORT DRIVE ERROR
4676
4677  006504  004737  015714      DRVER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4678  006510  104422  045006              DISPLY  ,EM30           ;REPORT DRIVE ERROR
4679  006514  004737  015760              JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4680  006520  004737  016332              JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4681  006524  004737  021270              JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4682  006530  004737  017534              JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4683  006534  000207                      RTS     PC              ;RETURN
4684
4685                                  ;PROCESS FORMAT ('FMT') ERROR
4686
4687  006536  032760  000400  000016  CKFMT:  BIT     #BIT8,$STATUS(R0) ;'HCRC' SET ON ORIGINAL ERROR ?
4688  006544  001402                      BEQ     .+6             ;BR IF NOT SET
4689  006546  000137  006354              JMP     HCRCER          ;REPORT HCRC ERROR
4690  006552  005046                      CLR     -(SP)           ;CLEAR STACK FOR DECREMENTED
4691  006554  005046                      CLR     -(SP)           ;SECTOR AND TRACK
4692  006556  004737  020036              JSR     PC,READDR       ;GET CORRECTED TRACK & SECTOR ADDRSSES
4693  006562  004737  012776              JSR     PC,READHD       ;READ HEADER
4694  006566  032737  000400  041472      BIT     #BIT8,GENREG+RHER1 ;'HCRC' SET WHEN HEADER READ?
4695  006574  001002                      BNE     .+6             ;BR IF 'HCRC' SET
4696  006576  000137  007546              JMP     FMTER           ;NO, ERROR IS 'FMT' ONLY
4697  006602  004737  015560      1$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT ON THE PACK
4698  006606  000207                      RTS     PC              ;RETURN IF IT IS
4699  006610  004737  015714              JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4700  006614  104422  044575              DISPLY  ,EM24           ;HEADER READ ERROR - FMT BIT DROPPED UP
4701  006620  004737  015760              JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4702  006624  004737  016332              JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4703  006630  004737  017070              JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
4704  006634  032760  040000  000174      BIT     #BIT14,$RHCS2(R0)  ;'WCE' ERROR ALSO ?
4705  006642  001402                      BEQ     .+6             ;BR IF NOT
4706  006644  004737  017170              JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
4707  006650  004737  017270              JSR     PC,LINE5A       ;DISPLAY HEADER
4708  006654  004737  021150              JSR     PC,INCSOF       ;INCREMENT SOFT ERROR COUNT
4709  006660  004737  021270              JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4710  006664  012760  000020  000024      MOV     #BIT4,$MASK(R0) ;SET ERROR MASK
4711  006672  012760  001400  000022      MOV     #1400,$RETRY(R0)  ;RETRY LIMIT
4712  006700  004737  013060              JSR     PC,RETRY        ;RETRY THE ORDER
4713  006704  000405                      BR      2$              ;RETRY NOT SUCESSFUL
4714  006706  004737  017474              JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
4715  006712  004737  017534              JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4716  006716  000207                      RTS     PC              ;RETURN
4717  006720  004737  017524      2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
4718  006724  004737  017534              JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
```

```
 4719  006730  004737  004222                    JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
 4720  006734  000207                            RTS     PC              ;RETURN
 4721
 4722                                    ;PROCESS HEADER COMPARE ('HCE') ERROR
 4723
 4724  006736  032760  000400  000200  CKHCE:  BIT     #BIT8,$RHER1(R0) ;HCRC SET ON ORIGINAL ERROR ?
 4725  006744  001402                            BEQ     .+6             ;BR IF NOT SET
 4726  006746  000137  006354                    JMP     HCRCER          ;REPORT HEADER CRC ERROR
 4727  006752  005046                    1$:     CLR     -(SP)           ;CLEAR STACK
 4728  006754  005046                            CLR     -(SP)           ;CLEAR STACK
 4729  006756  004737  020036                    JSR     PC,READDR       ;GET CURRENT SECTOR & TRACK ADDRS
 4730  006762  004737  012776                    JSR     PC,READHD       ;READ HEADER OF CURRENT SECTOR
 4731  006766  032737  000400  041472            BIT     #BIT8,GENREG+RHER1 ;'HCRC' SET
 4732  006774  001016                            BNE     2$              ;BR IF SET
 4733  006776  042737  010000  053554            BIC     #BIT12,CYLDER   ;CLEAR FORMAT BIT FROM HEADER
 4734  007004  026037  000222  053554            CMP     $RHCC(R0),CYLDER  ;CORRECT CYLINDER ?
 4735  007012  001402                            BEQ     .+6             ;BR IF IT IS
 4736  007014  000137  007166                    JMP     POSER           ;REPORT POSITIONING ERROR
 4737  007020  052737  010000  053554            BIS     #BIT12,CYLDER   ;RESTORE THE FORMAT BIT
 4738  007026  000137  007624                    JMP     HCEER           ;REPORT 'HCE' ERROR
 4739  007032  004737  015560            2$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
 4740  007036  000207                            RTS     PC              ;RETURN IF IT IS
 4741  007040  004737  015714                    JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 4742  007044  104422  044643                    DISPLY  ,EM25           ;HEADER READ ERROR - 'HCE' SET
 4743  007050  004737  015760                    JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
 4744  007054  004737  016332                    JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
 4745  007060  004737  017070                    JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
 4746  007064  032760  040000  000174            BIT     #BIT14,$RHCS2(R0)  ;'WCE' ERROR ALSO ?
 4747  007072  001402                            BEQ     .+6             ;BR IF NOT
 4748  007074  004737  017170                    JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
 4749  007100  004737  017270                    JSR     PC,LINE5A       ;PRINT LINE 5 OF ERROR MESSAGE
 4750  007104  004737  021150                    JSR     PC,INCSOF       ;INCREMENT SOFT ERROR COUNT
 4751  007110  004737  021270                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 4752  007114  012760  000200  000024            MOV     #BIT7,$MASK(R0) ;SET ERROR MASK
 4753  007122  012760  001400  000022            MOV     #1400,$RETRY(R0)  ;RETRY LIMIT
 4754  007130  004737  013060                    JSR     PC,RETRY        ;RETRY THE ORDER
 4755  007134  000405                            BR      3$              ;RETRY NOT SUCESSFUL
 4756  007136  004737  017474                    JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
 4757  007142  004737  017534                    JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
 4758  007146  000207                            RTS     PC              ;RETURN
 4759  007150  004737  017524            3$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
 4760  007154  004737  017534                    JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
 4761  007160  004737  004222                    JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
 4762  007164  000207                            RTS     PC              ;RETURN
 4763
 4764                                    ;REPORT POSSIBLE POSITIONING ERROR
 4765
 4766  007166  004737  012720            POSER:  JSR     PC,RECALT       ;RECALIBRATE
 4767  007172  004737  015714                    JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 4768  007176  104422  045722                    DISPLY  ,EM51           ;PROGRAM DETECTED POSITIONING ERROR
 4769  007202  004737  015760                    JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
 4770  007206  004737  016524                    JSR     PC,LINE3C       ;PRINT LINE 3C OF ERROR MESSAGE
 4771  007212  052737  010000  053554            BIS     #BIT12,CYLDER   ;RESTORE THE FORMAT BIT
 4772  007220  004737  017270                    JSR     PC,LINE5A       ;PRINT LINE 5A OF THE ERROR MESSAGE
```

```
 4773  007224  004737  021244                    JSR     PC,INCMIS       ;INCREMENT MISPOSITIONING COUNT
 4774  007230  004737  021270                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 4775  007234  004737  017662                    JSR     PC,LINE7A       ;PRINT LINE 7A OF ERROR MESSAGE
 4776  007240  000207                            RTS     PC              ;EXIT
 4777
 4778                                    ;REPORT 'OPI' ERROR
 4779
 4780  007242  004737  015560            OPIER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
 4781  007246  000207                            RTS     PC              ;RETURN IF IT IS
 4782  007250  004737  015714                    JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 4783  007254  104422  045040                    DISPLY  ,EM31           ;'OPI' ERROR
 4784  007260  004737  015760                    JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
 4785  007264  004737  016332                    JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
 4786  007270  004737  017070                    JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
 4787  007274  004737  021270                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 4788  007300  012760  020000  000024            MOV     #BIT13,$MASK(R0) ;ERROR MASK
 4789  007306  012760  001400  000022    OPIER1: MOV     #1400,$RETRY(R0) ;RETRY LIMIT
 4790  007314  004737  013060                    JSR     PC,RETRY        ;RETRY THE ORDER
 4791  007320  000405                            BR      1$              ;RETRY UNSUCESSFUL
 4792  007322  004737  017474                    JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
 4793  007326  004737  017534                    JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
 4794  007332  000207                            RTS     PC              ;EXIT
 4795  007334  004737  017524            1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
 4796  007340  004737  017534                    JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
 4797  007344  004737  004222                    JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
 4798  007350  000207                            RTS     PC              ;RETURN
 4799
 4800                                    ;REPORT 'DTE' ERROR
 4801
 4802  007352  004737  015560            DTEER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
 4803  007356  000207                            RTS     PC              ;RETURN IF IT IS
 4804  007360  004737  015714                    JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 4805  007364  104422  045103                    DISPLY  ,EM32           ;'DTE' ERROR
 4806  007370  000137  005426                    JMP     DCKER1          ;FINISH PROCESSING THE 'DTE' ERROR
 4807
 4808                                    ;REPORT 'PAR' ERROR
 4809
 4810  007374  004737  015714            PARER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 4811  007400  104422  045136                    DISPLY  ,EM33           ;REPORT 'PAR'
 4812  007404  004737  015760                    JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
 4813  007410  004737  016714                    JSR     PC,LINE3E       ;PRINT LINE 3E OF ERROR MESSAGE
 4814  007414  004737  017070                    JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
 4815  007420  004737  021270                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 4816  007424  012760  000010  000024            MOV     #BIT03,$MASK(R0) ;ERROR MASK
 4817  007432  012760  001400  000022            MOV     #1400,$RETRY(R0) ;RETRY LIMIT
 4818  007440  004737  013060                    JSR     PC,RETRY        ;RETRY ORDER
 4819  007444  000405                            BR      2$              ;RETRY UNSUCESSFUL
 4820  007446  004737  017474                    JSR     PC,LINE6C       ;RETRY SUCESSFUL
 4821  007452  004737  017534            1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
 4822  007456  000207                            RTS     PC              ;EXIT
 4823  007460  004737  017524            2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
 4824  007464  000772                            BR      1$              ;FINISH ERROR MESSAGE
 4825
 4826                                    ;REPORT 'IAE' ERROR
```

```
                                ;REPORT 'IAE' ERROR

007466  004737  015714        IAEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007472  104422  045255                DISPLY  .EM35           ;REPORT 'IAE'
007476  004737  015760                JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007502  004737  017016                JSR     PC,LINE3F       ;PRINT LINE 3F OF ERROR MESSAGE
007506  004737  021270                JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007512  004737  017534                JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007516  000207                      RTS     PC              ;RETURN

                                ;REPORT 'WLE' ERROR

007520  004737  015714        WLEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007524  104422  045313                DISPLY  .EM36           ;REPORT 'WLE'
007530  004737  015760                JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007534  004737  021270                JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007540  004737  017534                JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007544  000207                      RTS     PC              ;RETURN

                                ;REPORT FORMAT ERROR

007546  004737  015714        FMTER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007552  104422  044724                DISPLY  .EM26           ;FORMAT ERROR
007556  004737  015760                JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007562  004737  016332                JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
007566  004737  017070                JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
007572  032760  040000  000174        BIT     #BIT14,$RHCS2(R0)  ;'WCE' ERROR ALSO ?
007600  001402                      BEQ     .+6             ;BR IF NOT
007602  004737  017170                JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
007606  004737  017270                JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
007612  004737  021270                JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007616  004737  017534                JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007622  000207                      RTS     PC

                                ;REPORT HEACER COMPARE ERROR

007624  004737  015714        HCEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007630  104422  044751                DISPLY  .EM27           ;HEADER COMPARE ERROR
007634  004737  015760                JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007640  004737  016332                JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
007644  004737  017070                JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
007650  032760  040000  000174        BIT     #BIT14,$RHCS2(R0)  ;'WCE' ERROR ALSO ?
007656  001402                      BEQ     .+6             ;BR IF NOT
007660  004737  017170                JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
007664  004737  017270                JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
007670  004737  021270                JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007674  004737  017534                JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007700  004737  004222                JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
007704  000207                      RTS     PC              ;RETURN

                                ;PROCESS CONTROL/INTERFACE TRANSFER ERROR

007706  004737  015714        TRFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007712  104422  045344                DISPLY  .EM40           ;RH11 OR UNIBUS TRANSFER ERROR
007716  004737  015760                JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
```

```
4881  007722  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4882  007726  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
4883  007732  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4884  007736  032760  121400  000174          BIT     #BIT15!BIT13!BIT9!BIT8,$RHCS2(R0)  ;'DLT','UPE','MXF', 'MDPE' SET ?
4885  007744  001415                          BEQ     2$              ;BR IF NONE SET
4886  007746  012760  001400  000022          MOV     #1400,$RETRY(R0)  ;RETRY LIMIT
4887  007754  005060  000024                  CLR     $MASK(R0)       ;CLEAR ERROR MASK
4888  007760  004737  013060                  JSR     PC,RETRY        ;RETRY THE OPERATION
4889  007764  000403                          BR      1$              ;RETURN HERE IF RETRY UNSUCESSFUL
4890  007766  004737  017474                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
4891  007772  000402                          BR      2$              ;FINISH THE ERROR REPORT
4892  007774  004737  017524          1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
4893  010000  004737  017534          2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4894  010004  000207                          RTS     PC
4895
4896                                  ;PROCESS 'SKI' OR 'OCYL' ERRORS
4897
4898  010006  004737  015714          SKIER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4899  010012  104422  045654                  DISPLY  ,EM50           ;'SKI' OR 'OCYL' ERROR
4900  010016  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4901  010022  004737  016512                  JSR     PC,LINE3B       ;PRINT LINE 3B OF ERROR MESSAGE
4902  010026  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4903  010032  004737  021220                  JSR     PC,INCSKI       ;INCREMENT 'SKI' OR 'OCYL' ERROR COUNT
4904  010036  004737  017662                  JSR     PC,LINE7A       ;PRINT LINE 7A OF ERROR MESSAGE
4905  010042  000207                          RTS     PC
4906
4907                                  ;REPORT WRITE CLOCK FAILURE ('WCF')
4908
4909  010044  004737  015714          WCFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4910  010050  104422  045213                  DISPLY  ,EM34           ;REPORT WRITE CLOCK FAILURE
4911  010054  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4912  010060  004737  016340                  JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
4913  010064  004737  017070                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
4914  010070  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4915  010074  004737  012522                  JSR     PC,PRTBAD       ;SEE IF BAD SECTOR TO BE PRINTED
4916  010100  012760  001400  000022          MOV     #1400,$RETRY(R0)  ;RETRY COUNT
4917  010106  012760  000040  000024          MOV     #BIT05,$MASK(R0)  ;ERROR MASK
4918  010114  004737  013060                  JSR     PC,RETRY        ;RETRY THE ORDER
4919  010120  000405                          BR      2$              ;RETURN HERE IF RETRY UNSUCESSFUL
4920  010122  004737  017474                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
4921  010126  004737  017534          1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4922  010132  000207                          RTS     PC
4923  010134  004737  017524          2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
4924  010140  000772                          BR      1$
4925
4926                                  ;PROCESS DRIVE UNSAFE ERROR
4927
4928  010142  004737  015714          UNSAF:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
4929  010146  104422  045765                  DISPLY  ,EM60           ;REPORT DRIVE UNSAFE
4930  010152  004737  015760                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
4931  010156  004737  016332                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
4932  010162  004737  021270                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
4933  010166  004737  017534                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
4934  010172  000207                          RTS     PC
```

```
                                        ;REPORT AN 'UNKNOWN' DATA PATTERN
                                        ;
010174  105737  011321      NOMTCH: TSTB    FRSTER          ;FIRST ERROR IN THE SECTOR ?
010200  001013                      BNE     1$              ;BR IF NOT OR IF PROCESSING 'DCKER'
010202  004737  015714              JSR     PC,LINE1        ;TYPE LINE 1 OF ERROR MESSAGE
010206  104422  045523              DISPLY  ,EM43           ;'CAN'T MATCH DATA WITH PATTERN'
010212  004737  015760              JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
010216  004737  016340              JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
010222  004737  017070              JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
010226  000404                      BR      2$              ;CONTINUE PROCESSING ERROR
010230  104422  045523      1$:     DISPLY  ,EM43           ;'CAN'T MATCH DATA WITH PATTERN'
010234  104422  001157              DISPLY  ,$CRLF          ;CR-LF
010240  104422  050611      2$:     DISPLY  ,LIN9I          ;HEADER FOR DATA PRINTOUT
010244  010146                      MOV     R1,-(SP)        ;ADDRESS OF WORD 1
(1) 010246  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010252  104422  051274          DISPLY  ,LINSP          ;SPACES
(1) 010256  012146                  MOV     (R1)+,-(SP)     ;WORD 1
(1) 010260  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010264  104422  001157          DISPLY  ,$CRLF          ;CR-LF
(1) 010270  010146                  MOV     R1,-(SP)        ;ADDRESS OF WORD 2
(1) 010272  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010276  104422  051274          DISPLY  ,LINSP          ;SPACES
(1) 010302  012146                  MOV     (R1)+,-(SP)     ;WORD 2
(1) 010304  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010310  104422  001157          DISPLY  ,$CRLF          ;CR-LF
(1) 010314  010146                  MOV     R1,-(SP)        ;ADDRESS OF WORD 3
(1) 010316  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010322  104422  051274          DISPLY  ,LINSP          ;SPACES
(1) 010326  012146                  MOV     (R1)+,-(SP)     ;WORD 3
(1) 010330  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010334  104422  001157          DISPLY  ,$CRLF          ;CR-LF
(1) 010340  010146                  MOV     R1,-(SP)        ;ADDRESS OF WORD 4
(1) 010342  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010346  104422  051274          DISPLY  ,LINSP          ;SPACES
(1) 010352  012146                  MOV     (R1)+,-(SP)     ;WORD 4
(1) 010354  004737  024652          JSR     PC,PRTOCT       ;PRINT IT
(1) 010360  104422  001157          DISPLY  ,$CRLF          ;CR-LF
4957 010364  062701  000770              ADD     #770,R1         ;INCREMENT BUFFER POINTER
4958 010370  060237  011326              ADD     R2,ERCTR        ;ADD SECTOR (OR LESS) COUNT TO THE
4959                                                         ;COMPARSION ERROR COUNT
4960 010374  005002                      CLR     R2              ;CLEAR 'WORDS TO COMPARE' COUNT IN R2
4961 010376  112737  177777  011321      MOVB    #-1,FRSTER      ;SET 'NOT FIRST ERROR' INDICATOR
4962 010404  000207                      RTS     PC              ;RETURN
4963
4964                                    ;CHECK ERROR BITS IN THE RH11 & RP04 REGISTERS
4965
4966 010406  032760  060000  000164  CKERR:  BIT     #60000,$RHCS1(R0)  ;SEE IF 'TRE' OR 'MCPE' SET
4967 010414  001015                      BNE     1$              ;BR IT EITHER SET
4968 010416  032760  177400  000174      BIT     #177400,$RHCS2(R0)  ;SEE IF ERROR BITS IN CS2 SET
4969 010424  001011                      BNE     1$              ;BR IF ANY SET
4970 010426  005760  000200              TST     $RHER1(R0)      ;ANY BITS SET IN ER1
4971 010432  001006                      BNE     1$              ;BR IF ANY SET
4972 010434  005760  000224              TST     $RHER2(R0)      ;ANY BITS SET IN ER2 ?
```

```
4973  010440  001003                    BNE     1$                ;BR IF ANY SET
4974  010442  005760  000226            TST     $RHER3(R0)        ;ANY BITS SET IN ER3 ?
4975  010446  001416                    BEQ     2$                ;BR IF NONE SET
4976  010450  004737  015714      1$:   JSR     PC,LINE1          ;PRINT LINE 1 OF ERROR MESSAGE
4977  010454  104422  045570            DISPLY  ,EM44             ;ERROR BITS SET, BUT 'SC' OR 'TRE' NOT SET
4978  010460  004737  015760            JSR     PC,LINE2          ;PRINT LINE 2 OF ERROR MESSAGE
4979  010464  004737  016332            JSR     PC,LINE3          ;PRINT LINE 3 OF ERROR MESSAGE
4980  010470  004737  017070            JSR     PC,LINE4          ;PRINT LINE 4 OF ERROR MESSAGE
4981  010474  004737  021270            JSR     PC,INCTOT         ;INCREMENT TOTAL ERROR COUNT
4982  010500  004737  017534            JSR     PC,LINE7          ;PRINT LINE 7 OF ERROR MESSAGE
4983  010504  000207              2$:   RTS     PC                ;RETURN
4984
4985                                    ;CHECK BUS ADDRESS REGISTER & WORD COUNT REGISTER
4986
4987  010506  005760  000166      CKBUS: TST    $RHWC(R0)         ;CHECK WORD COUNT
4988  010512  001010                    BNE     1$                ;BR IF NOT ZERO
4989  010514  016046  000020            MOV     $WRDL(R0),-(SP)   ;WORD LENGTH
4990  010520  006316                    ASL     (SP)              ;CHANGE INTO BYTE COUNT
4991  010522  066016  000006            ADD     $BUF(R0),(SP)     ;ADD THE STARTING LOCATION
4992  010526  022660  000170            CMP     (SP)+,$RHBA(R0)   ;BUFFER ADDRESS PROPER ?
4993  010532  001416                    BEQ     2$                ;BR IF OK
4994  010534  004737  015714      1$:   JSR     PC,LINE1          ;PRINT LINE 1 OF ERROR MESSAGE
4995  010540  104422  045402            DISPLY  ,EM41             ;BUS ADDRESS OR WORD COUNT INCORRECT
4996  010544  004737  015760            JSR     PC,LINE2          ;PRINT LINE 2 OF ERROR MESSAGE
4997  010550  004737  016646            JSR     PC,LINE3D         ;PRINT LINE 3D OF ERROR MESSAGE
4998  010554  004737  017070            JSR     PC,LINE4          ;PRINT LINE 4 OF ERROR MESSAGE
4999  010560  004737  021270            JSR     PC,INCTOT         ;INCREMENT TOTAL ERROR COUNT
5000  010564  004737  017534            JSR     PC,LINE7          ;PRINT LINE 7 OF ERROR MESSAGE
5001  010570  000207              2$:   RTS     PC
5002
5003                                    ;COMPARE THE BUFFER
5004
5005  010572  132760  000004  000030 CMPAR: BITB    #BIT02,$CODE(R0)  ;SEE IF READ ORDER
5006  010600  001001                    BNE     .+4               ;BR IF IT IS
5007  010602  000207                    RTS     PC                ;RETURN
5008  010604  032737  000002  177570    BIT     #SW01,SWR         ;IS SWITCH 1 SET?
5009  010612  001401                    BEQ     .+4               ;BR IF NOT
5010  010614  000207                    RTS     PC                ;YES, DON'T COMPARE
5011  010616  105037  011321            CLRB    FRSTER            ;CLEAR 'FIRST ERROR' INDICATOR
5012  010622  005037  011326      CMPARD: CLR   ERCTR             ;CLEAR THE ERROR COUNTER
5013  010626  016001  000006            MOV     $BUF(R0),R1       ;BUFFER ADDRESS
5014  010632  016037  000020  011332    MOV     $WRDL(R0),CMCNT   ;WORD COUNT TO WORKING LOCATION
5015  010640  066037  000166  011332    ADD     $RHWC(R0),CMCNT   ;CALCULATE ACTUAL WORDS TRANSFERED
5016  010646  016037  000012  011334    MOV     $CYL(R0),CMCYL    ;CYLINDER ADDRESS WORKING LOCATION
5017  010654  052737  010000  011334    BIS     #BIT12,CMCYL      ;SET FORMAT BIT
5018  010662  016037  000010  011336    MOV     $SEC(R0),CMSEC    ;SECTOR & TRACK ADDRESSES TO WORKING LOCNS
5019  010670  013737  001300  011330    MOV     CMPLMT,LIMIT      ;DISPLAY LIMIT
5020  010676  005237  011330            INC     LIMIT             ;CONVERT PARAMETER INTO LIMIT VALUE
5021  010702  112737  177777  011320 CMSTR: MOVB    #-1,ZROIND        ;CLEAR THE 'ZERO'S' INDICATOR
5022  010710  005037  011322            CLR     SAVER1            ;CLEAR THE R1 SAVE WORD
5023  010714  005037  011324            CLR     SAVER5            ;CLEAR THE R5 SAVE WORD
5024  010720  023760  011332  000026    CMP     CMCNT,$SSEC(R0)   ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
5025  010726  101003                    BHI     .+10              ;--)  BR IF IT IS
5026  010730  013702  011332            MOV     CMCNT,R2          ;  I  LESS THAN, USE REMAINING BUFFER
```

```
5027  010734  000402                        BR      .+6             ;--I->
5028  010736  016002  000026                MOV     $SSEC(R0),R2    ;<-- I  COMPARE SECTOR
5029  010742  166037  000026  011332        SUB     $SSEC(R0),CMCNT ;<----  DECREMENT WORD COUNT
5030  010750  126027  000030  000005        CMPB    $CODE(R0),#5    ;READ HEADER & DATA?
5031  010756  001036                        BNE     CMDAT           ;BR IF NOT
5032  010760  023721  011334        CMHED:  CMP     CMCYL,(R1)+     ;CHECK CYLINDER
5033  010764  001402                        BEQ     .+6             ;BR IF COMPARE OK
5034  010766  004737  011042                JSR     PC,1$           ;REPORT ERROR
5035  010772  023721  011336                CMP     CMSEC,(R1)+     ;COMPARE SECTOR & TRACK
5036  010776  001402                        BEQ     .+6             ;BR IF EQ
5037  011000  004737  011042                JSR     PC,1$           ;REPORT ERROR
5038  011004  005721                        TST     (R1)+           ;1ST KEY WORD ZERO?
5039  011006  001402                        BEQ     .+6             ;BR IF IT IS
5040  011010  004737  011042                JSR     PC,1$           ;REPORT ERROR
5041  011014  005721                        TST     (R1)+           ;CHECK 2ND KEY WORD
5042  011016  001402                        BEQ     .+6             ;BR IF ZERO
5043  011020  004737  011042                JSR     PC,1$           ;REPORT THE ERROR
5044  011024  162702  000004                SUB     #4,R2           ;SUBTRACT HEADER LENGTH FROM SIZE
5045  011030  003530                        BLE     CMPRX           ;BR IF FINISHED
5046  011032  022702  000004                CMP     #4,R2           ;SEE IF AT LEAST 4 MORE WORDS TO CHECK
5047  011036  101125                        BHI     CMPRX           ;BR IF NOT
5048  011040  000405                        BR      CMDAT           ;COMPARE THE DATA PORTION
5049  011042  005237  011326        1$:     INC     ERCTR           ;INCREMENT THE ERROR COUNT
5050  011046  004737  011340                JSR     PC,CMPRT        ;REPORT THE COMPARISON ERROR
5051  011052  000207                        RTS     PC              ;CHECK THE REST OF THE HEADER
5052  011054  004737  011660        CMDAT:  JSR     PC,MATCH        ;FIND THE PATTERN
5053  011060  000403                        BR      2$              ;FOUND A PATTERN
5054  011062  004737  010174                JSR     PC,NOMTCH       ;RETURN HERE IF NO MATCH WITH PATTERN MADE
5055  011066  000456                        BR      8$              ;BYPASS COMPARE ROUTINE
5056  011070  011405                2$:     MOV     (R4),R5         ;ADDRESS OF PATTERN ADDRESS IN R4
5057  011072  012703  000020                MOV     #20,R3          ;R3 IS PATTERN POS COUNTER
5058  011076  022125                3$:     CMP     (R1)+,(R5)+     ;COMPARE BUFFER WITH PATTERN
5059  011100  001016                        BNE     5$              ;BR IF NOT EQUAL
5060  011102  005737  011326                TST     ERCTR           ;ERRORS DETECTED ?
5061  011106  001406                        BEQ     4$              ;BR IF NO ERRORS
5062  011110  032737  000010  177570        BIT     #SW3,SWR        ;SWITCH 3 SET ?
5063  011116  001402                        BEQ     4$              ;BR IF NOT SET
5064  011120  004737  011340                JSR     PC,CMPRT        ;DISPLAY THE WORD
5065  011124  005302                4$:     DEC     R2              ;DECREMENT SIZE COUNT
5066  011126  001436                        BEQ     8$              ;BR WHEN AT END
5067  011130  005303                        DEC     R3              ;DECREMENT PATT POS COUNT
5068  011132  001361                        BNE     3$              ;BR IF NOT AT END OF PATT
5069  011134  000755                        BR      2$              ;RESTART THE PATTERN
5070  011136  005761  177776        5$:     TST     -2(R1)          ;IS MISCOMPARED CHARACTER=0
5071  011142  001410                        BEQ     6$              ;BR IF YES
5072  011144  112737  177777  011320        MOVB    #-1,ZROIND      ;SET NON-ZERO MISCOMPARED IND
5073  011152  005237  011326                INC     ERCTR           ;INCREMENT THE ERROR COUNTER
5074  011156  004737  011340                JSR     PC,CMPRT        ;REPORT ERROR
5075  011162  000760                        BR      4$              ;CONTINUE COMPARE
5076  011164  105737  011321        6$:     TSTB    FRSTER          ;FIRST ERROR?
5077  011170  100407                        BMI     7$              ;BR IF NOT
5078  011172  105037  011320                CLRB    ZROIND          ;SET THE ZERO INDICATOR
5079  011176  010137  011322                MOV     R1,SAVER1       ;SAVE CURRENT R1
5080  011202  010537  011324                MOV     R5,SAVER5       ;SAVE CURRENT R5
```

```
5081  011206  000746                    BR      4$              ;CONTINUE COMPARE
5082  011210  105737  011320    7$:     TSTB    ZROIND          ;ANY MISCOMPARIONS NOT ZEROS ?
5083  011214  001743                    BEQ     4$              ;BR IF NONE-ALL ERRORS=ZERO
5084  011216  004737  011340            JSR     PC,CMPRT        ;REPORT ERROR
5085  011222  000740                    BR      4$              ;CONTINUE COMPARING
5086  011224  005737  011332    8$:     TST     CMCNT           ;AT END OF BUFFER
5087  011230  003430                    BLE     CMPRX           ;BR IF AT END
5088  011232  126027  000030  000005    CMPB    $CODE(R0),#5    ;SEE IF READ HEADER & DATA
5089  011240  001220                    BNE     CMSTR           ;BR IF NOT
5090  011242  105237  011336            INCB    CMSEC           ;INCREMENT SECTOR
5091  011246  123727  011336  000026    CMPB    CMSEC,#22.      ;SECTOR GREATER THAN MAX ?
5092  011254  103612                    BLO     CMSTR           ;BR IF NOT GREATER THAN MAX
5093  011256  105037  011336            CLRB    CMSEC           ;CLEAR SECTOR ADDRESS
5094  011262  105237  011337            INCB    CMTRK           ;INCREMENT TRACK
5095  011266  123727  011337  000023    CMPB    CMTRK,#19.      ;TRACK GREATER THAN MAX ?
5096  011274  103602                    BLO     CMSTR           ;BR IF NOT GREATER
5097  011276  105037  011337            CLRB    CMTRK           ;RESET TRACK ADDRESS
5098  011302  005237  011334            INC     CMCYL           ;INCREMENT CYLINDER ADDRESS
5099  011306  000137  010702            JMP     CMSTR           ;CONTINUE WITH COMPARE
5100  011312  004737  011614  CMPRX:    JSR     PC,ENDCMP       ;PRINT LAST LINE IF ERRORS
5101  011316  000207                    RTS     PC
5102
5103  011320     377  ZROIND: .BYTE   -1              ;ZERO INDICATOR
5104  011321     000  FRSTER: .BYTE   0               ;FIRST ERROR INDICATOR
5105                                                   ;IF > 0, PROCESSING 'DCKER'
5106                                                   ;IF < 0, MISCOMPARSION FOUND
5107  011322  000000  SAVER1: .WORD   0               ;SAVE R1 HERE
5108  011324  000000  SAVER5: .WORD   0               ;SAVE R5 HERE
5109  011326  000000  ERCTR:  .WORD   0               ;NUMBER OF ERRORS
5110  011330  000000  LIMIT:  .WORD   0               ;DISPLAY LIMIT
5111  011332  000000  CMCNT:  .WORD   0               ;WORD COUNT
5112  011334  000000  CMCYL:  .WORD   0               ;CYLINDER ADDRESS
5113  011336     000  CMSEC:  .BYTE   0               ;SECTOR ADDRESS
5114  011337     000  CMTRK:  .BYTE   0               ;TRACK ADDRESS
5115
5116                  ;TYPE DATA COMPARE ERRORS
5117
5118  011340  005737  011322  CMPRT:  TST     SAVER1          ;PRINT SAVED VALUES ?
5119  011344  001010                  BNE     2$              ;BR IF NOT
5120  011346  105737  011321          TSTB    FRSTER          ;FIRST ERROR?
5121  011352  100402                  BMI     1$              ;BR IF NOT
5122  011354  004737  011434          JSR     PC,4$           ;PRINT INITIAL MESSAGE INFO
5123  011360  004737  011516  1$:     JSR     PC,8$           ;PRINT REMAINDER OF MESSAGE
5124  011364  000422                  BR      3$              ;EXIT
5125  011366                  2$:
 (2)  011366  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (2)  011370  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
5126  011372  013701  011322          MOV     SAVER1,R1       ;DISPLAY SAVED R1
5127  011376  013705  011324          MOV     SAVER5,R5       ;DISPLAY SAVED R5
5128  011402  004737  011434          JSR     PC,4$           ;PRINT INITIAL MESSAGE INFO
5129  011406  004737  011516          JSR     PC,8$           ;PRINT SAVED VALUES
5130  011412  005037  011322          CLR     SAVER1          ;CLEAR SAVED REGISTER INDICATORS
5131  011416  005037  011324          CLR     SAVER5          ;CLEAR THE OTHER ONE
5132  011422  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
```

```
  (2)  011424  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 5133  011426  004737  011516                JSR     PC,8$           ;PRINT REMAINDER OF MESSAGE
 5134  011432  000207                3$:     RTS     PC              ;RETURN
 5135  011434  105737  011321        4$:     TSTB    FRSTER          ;FIRST ERROR ?
 5136  011440  100425                        BMI     7$              ;BR IF NOT
 5137  011442  001013                        BNE     5$              ;BR IF FIRST ERROR AND PROCESSING 'DCK' ERROR
 5138  011444  004737  015714                JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
 5139  011450  104422  045446                DISPLY  ,EM42           ;DATA COMPARE ERROR
 5140  011454  004737  015760                JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
 5141  011460  004737  016340                JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
 5142  011464  004737  017070                JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
 5143  011470  000404                        BR      6$
 5144  011472  104422  050504        5$:     DISPLY  ,LIN9B          ;HEADER MESSAGE OF PROCESSING 'DCK' ERROR
 5145  011476  104422  001157                DISPLY  ,$CRLF
 5146  011502  104422  050533        6$:     DISPLY  ,LIN9H          ;DISPLAY HEADER
 5147  011506  112737  177777  011321        MOVB    #-1,FRSTER      ;SET FIRST ERROR INDICATOR
 5148  011514  000207                7$:     RTS     PC              ;RETURN
 5149  011516  005737  011330        8$:     TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
 5150  011522  001403                        BEQ     9$              ;BR IF IT HAS
 5151  011524  005337  011330                DEC     LIMIT           ;DECREMENT LIMIT COUNTER
 5152  011530  001005                        BNE     10$             ;BR IF NOT AT LIMIT
 5153  011532  032737  000200  177570  9$:     BIT     #SW07,SWR       ;PRINT ALL DATA COMPARE ERRORS ?
 5154  011540  001001                        BNE     10$             ;BR IF YES
 5155  011542  000207                        RTS     PC              ;RETURN
 5156  011544  010146                10$:     MOV     R1,-(SP)        ;BUFFER ADDRESS
 5157  011546  162716  000002                SUB     #2,(SP)         ;ADJUST ADDRESS
 5158  011552  004737  024652                JSR     PC,PRTOCT       ;PRINT IT
 5159  011556  104422  051274                DISPLY  ,LINSP          ;2 SPACES
 5160  011562  016546  177776                MOV     -2(R5),-(SP)    ;GOOD DATA
 5161  011566  004737  024652                JSR     PC,PRTOCT       ;PRINT IT
 5162  011572  104422  051274                DISPLY  ,LINSP          ;2 SPACES
 5163  011576  016146  177776                MOV     -2(R1),-(SP)    ;BAD DATA
 5164  011602  004737  024652                JSR     PC,PRTOCT       ;PRINT IT
 5165  011606  104422  001157                DISPLY  ,$CRLF          ;CR-LF
 5166  011612  000207                        RTS     PC              ;RETURN
 5167
 5168                                  ;LAST LINE OF COMPARE ERROR REPORTING
 5169
 5170  011614  005737  011326        ENDCMP: TST     ERCTR           ;SEE IF ANY ERRORS
 5171  011620  001416                        BEQ     1$              ;BR IF NOT
 5172  011622  104422  050631                DISPLY  ,LIN9E          ;'NUMBER OF ERRORS='
 5173  011626  013746  011326                MOV     ERCTR,-(SP)     ;NUMBER OF ERRORS
 5174  011632  004737  030170                JSR     PC,$S82D        ;CONVERT IT
 5175  011636  004737  025510                JSR     PC,$SUPRS       ;PRINT IT
 5176  011642  104422  001157                DISPLY  ,$CRLF          ;CR-LF
 5177  011646  004737  021270                JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
 5178  011652  004737  017534                JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
 5179  011656  000207                1$:     RTS     PC
 5180
 5181
 5182                                  ;FIND THE CORRECT PATTERN - RETURN WITH ADDRESS OF PATTERN IN R4
 5183                                  ;       RETURN +2 IF PATTERN CAN'T BE FOUND
 5184
 5185  011660  010146                MATCH:  MOV     R1,-(SP)        ;SAVE R1 ON THE STACK
```

```
 5186  011662  012704  000044                    MOV     #44,R4          ;PATTERN TABLE INDEX
 5187  011666  011601                    1$:     MOV     (SP),R1         ;RELOAD R1
 5188  011670  162704  000002                    SUB     #2,R4           ;DECREMENT INDEX
 5189  011674  016405  042556                    MOV     STNDAT(R4),R5   ;ADDRESS OF PATTERN ADDRESS
 5190  011700  001411                            BEQ     3$              ;BR IF ALL PATTERNS CHECKED AND NO MATCH
 5191                                                                    ;FOUND
 5192  011702  012703  000004                    MOV     #4,R3           ;NUMBER OF LOCATIONS TO CHECK
 5193  011706  022125                    2$:     CMP     (R1)+,(R5)+     ;COMPARE THE BUFFER AGAINST THE PATTERN
 5194  011710  001366                            BNE     1$              ;BR IF NOT EQUAL, TRY NEXT PATTERN
 5195  011712  005303                            DEC     R3              ;FINISHED CHECKING?
 5196  011714  001374                            BNE     2$              ;BR IF NOT FINISHED
 5197  011716  062704  042556                    ADD     #STNDAT,R4      ;MAKE PATTERN ADDRESS ABSOLUTE
 5198  011722  000403                            BR      4$              ;EXIT
 5199  011724  062766  000002  000002    3$:     ADD     #2,2(SP)        ;INCREMENT RETURN ADDRESS
 5200  011732  012601                    4$:     MOV     (SP)+,R1        ;RESTORE R1
 5201  011734  000207                            RTS     PC              ;RETURN
 5202
 5203                                    ;USE ECC TO CORRECT THE DATA ERROR
 5204
 5205  011736  016037  000170  012500    ECC:    MOV     $RHBA(R0),ECSEC ;ADDRESS OF LAST LOCN XFERED
 5206  011744  016046  000166                    MOV     $RHWC(R0),-(SP) ;ACT WORDS XFERED (2'S COMP)
 5207  011750  066016  000020                    ADD     $WRDL(R0),(SP)  ;ADD WORDS REQUESTED
 5208  011754  005046                            CLR     -(SP)           ;CLEAR NEXT STACK LOCN
 5209  011756  016046  000026                    MOV     $SSEC(R0),-(SP) ;SECTOR SIZE
 5210  011762  004737  024136                    JSR     PC,LINKDV       ;DIVIDE WORDS XFERED BY SECTOR SIZE
 5211  011766  005716                            TST     (SP)            ;PARTIAL SECTOR XFERED ?
 5212  011770  001413                            BEQ     1$              ;BR IF NOT
 5213  011772  006316                            ASL     (SP)            ;CONVERT INTO NUMBER OF BYTES
 5214  011774  161637  012500                    SUB     (SP),ECSEC      ;SUBTRACT SECTOR RESIDUE
 5215  012000  126027  000030  000005            CMPB    $CODE(R0),#5    ;WAS OP READ HEAD & DATA
 5216  012006  001007                            BNE     2$              ;BR IF NOT
 5217  012010  062737  000010  012500            ADD     #8.,ECSEC       ;ADD HEADER SIZE (IN BYTES) BACK IN
 5218  012016  000403                            BR      2$              ;GO ADJUST THE STACK POINTER
 5219  012020  162737  001000  012500  1$:     SUB     #1000,ECSEC     ;SUBTRACT SECTOR DATA FIELD SIZE (IN BYTES)
 5220  012026  062706  000004            2$:     ADD     #4,SP           ;ADJUST THE STACK POINTER
 5221  012032  016037  000230  012476            MOV     $RHEC1(R0),ECBIT  ;ECC POSITION COUNT
 5222  012040  005337  012476                    DEC     ECBIT           ;ADJUST THE POSITION COUNT
 5223  012044  013737  012476  012506            MOV     ECBIT,ECWRD     ;LOAD THE WORD COUNT LOCATION
 5224  012052  042737  177760  012476            BIC     #^C17,ECBIT     ;SAVE THE BIT OFFSET COUNT
 5225  012060  042737  000017  012506            BIC     #17,ECWRD       ;CLEAR THE BIT OFFSET
 5226  012066  006237  012506                    ASR     ECWRD           ;CHANGE TO BYTE COUNT
 5227  012072  006237  012506                    ASR     ECWRD           ;CHANGE TO BYTE COUNT
 5228  012076  006237  012506                    ASR     ECWRD           ;CHANGE TO BYTE COUNT
 5229  012102  104422  050710                    DISPLY  ,LIN10A         ;'ERROR BURST BEGINS AT '
 5230  012106  013746  012506                    MOV     ECWRD,-(SP)     ;PUT THE WORD COUNT ON THE STACK
 5231  012112  006216                            ASR     (SP)            ;CONVERT TO WORD COUNT FOR MESSAGE
 5232  012114  004737  030170                    JSR     PC,$SB2D        ;CONVERT THE WORD COUNT
 5233  012120  004737  025510                    JSR     PC,$SUPRS       ;PRINT IT
 5234  012124  104422  050744                    DISPLY  ,LIN10B         ;' IN DATA FIELD OF ERROR SECTOR'
 5235  012130  063737  012500  012506            ADD     ECSEC,ECWRD     ;FIND THE BEGINNING OF THE ERROR BURST
 5236  012136  026037  000170  012506            CMP     $RHBA(R0),ECWRD  ;SEE IF BURST WAS IN DATA READ
 5237  012144  101002                            BHI     .+6             ;BR IF IN DATA READ
 5238  012146  000137  012464                    JMP     ECC2            ;NOT IN DATA READ - REPORT IT
 5239  012152  016037  000232  012502            MOV     $RHEC2(R0),ECMSK0  ;GET THE ERROR MASK
```

```
 5240  012160  005037  012504                CLR     ECMSK1          ;CLEAR THE UPPER MASK WORD
 5241  012164  005737  012476          3$:   TST     ECBIT           ;BIT OFFSET EQUAL ZERO
 5242  012170  001407                        BEQ     4$              ;BR IF IT IS
 5243  012172  005337  012476                DEC     ECBIT           ;DECREMENT THE BIT OFFSET COUNT
 5244  012176  006337  012502                ASL     ECMSK0          ;SHIFT THE ERROR MASK
 5245  012202  006137  012504                ROL     ECMSK1          ;SHIFT THE LOWER INTO THE UPPER
 5246  012206  000766                        BR      3$              ;CONTINUE THE SHIFT
 5247  012210  017737  000272  012512  4$:   MOV     @ECWRD,ECBAD0   ;SAVE THE INCORRECT WORD
 5248  012216  005037  012514                CLR     ECWRD1          ;CLEAR SECOND INCORRECT WORD ADDRESS
 5249  012222  013746  012502                MOV     ECMSK0,-(SP)    ;PUT LOWER MASK ON STACK
 5250  012226  047716  000254                BIC     @ECWRD,(SP)     ;CLEAR ERRONEOUS ONE BITS FROM MASK
 5251  012232  043777  012502  000246        BIC     ECMSK0,@ECWRD   ;CLEAR ERRONEOUS ONE BITS FROM BAD WORD
 5252  012240  052677  000242                BIS     (SP)+,@ECWRD    ;SET DROPPED BITS
 5253  012244  005737  012504                TST     ECMSK1          ;DOES BURST GO INTO NEXT WORD ?
 5254  012250  001431                        BEQ     ECC1            ;BR IF BURST ONLY IN ONE WORD
 5255  012252  013737  012506  012514        MOV     ECWRD,ECWRD1    ;DUPLICATE ADDRESS
 5256  012260  062737  000002  012514        ADD     #2,ECWRD1       ;INCREMENT ERROR ADDRESS
 5257  012266  026037  000170  012514        CMP     $RHBA(R0),ECWRD1  ;IS NEXT WORD IN THE BUFFER
 5258  012274  101003                        BHI     5$              ;BR IF IT IS
 5259  012276  005037  012514                CLR     ECWRD1          ;CLEAR 2ND WORD ADDRESS
 5260  012302  000414                        BR      ECC1            ;PRINT WORD CORRECTED
 5261  012304  017737  000204  012520  5$:   MOV     @ECWRD1,ECBAD1  ;SAVE THE SECOND BAD WORD
 5262  012312  013746  012504                MOV     ECMSK1,-(SP)    ;PUT THE UPPER MASK ON THE STACK
 5263  012316  047716  000172                BIC     @ECWRD1,(SP)    ;CLEAR ERRONEOUS ONE BITS FROM UPPER MASK
 5264  012322  043777  012504  000164        BIC     ECMSK1,@ECWRD1  ;CLEAR ERRONEOUS ONE BITS FROM DATA WORD
 5265  012330  052677  000160                BIS     (SP)+,@ECWRD1   ;SET DROPPED BITS
 5266  012334  104422  051112          ECC1: DISPLY  ,LIN10H         ;HEADER
 5271  012340  013746  012506                MOV     ECWRD,-(SP)     ;PUT ECWRD ON THE STACK
  (1)  012344  004737  024652                JSR     PC,PRTOCT       ;PRINT ECWRD
  (1)  012350  104422  051274                DISPLY  ,LINSP          ;SPACES
  (1)  012354  013746  012512                MOV     ECBAD0,-(SP)    ;PUT ECBAD0 ON THE STACK
  (1)  012360  004737  024652                JSR     PC,PRTOCT       ;PRINT ECBAD0
  (1)  012364  104422  051274                DISPLY  ,LINSP          ;SPACES
  (1)  012370  017746  000112                MOV     @ECWRD,-(SP)    ;PUT @ECWRD ON THE STACK
  (1)  012374  004737  024652                JSR     PC,PRTOCT       ;PRINT @ECWRD
  (1)  012400  104422  051274                DISPLY  ,LINSP          ;SPACES
 5272  012404  005737  012514                TST     ECWRD1          ;PRINT THE NEXT WORD ?
 5273  012410  001427                        BEQ     ECCX            ;BR IF NOT
 5274  012412  104422  001157                DISPLY  ,$CRLF          ;CR-LF
 5279  012416  013746  012514                MOV     ECWRD1,-(SP)    ;PUT ECWRD1 ON THE STACK
  (1)  012422  004737  024652                JSR     PC,PRTOCT       ;PRINT ECWRD1
  (1)  012426  104422  051274                DISPLY  ,LINSP          ;SPACES
  (1)  012432  013746  012520                MOV     ECBAD1,-(SP)    ;PUT ECBAD1 ON THE STACK
  (1)  012436  004737  024652                JSR     PC,PRTOCT       ;PRINT ECBAD1
  (1)  012442  104422  051274                DISPLY  ,LINSP          ;SPACES
  (1)  012446  017746  000042                MOV     @ECWRD1,-(SP)   ;PUT @ECWRD1 ON THE STACK
  (1)  012452  004737  024652                JSR     PC,PRTOCT       ;PRINT @ECWRD1
  (1)  012456  104422  051274                DISPLY  ,LINSP          ;SPACES
 5280  012462  000402                        BR      ECCX            ;EXIT
 5281  012464  104422  051005          ECC2: DISPLY  ,LIN10C         ;ERROR BURST WAS NOT TRANSFERED TO MEMORY
 5282  012470  104422  001157          ECCX: DISPLY  ,$CRLF          ;CR-LF
 5283  012474  000207                        RTS     PC              ;RETURN
 5284
 5285  012476  000000                ECBIT:  .WORD   0               ;ERROR BURST BIT OFFSET
```

```
5286  012500  000000                ECSEC:  .WORD   0               ;ERROR BURST WORD OFFSET (RELATIVE TO SECTOR)
5287  012502  000000                ECMSK0: .WORD   0               ;CORRECTION MASK FOR FIRST ERROR WORD
5288  012504  000000                ECMSK1: .WORD   0               ;CORRECTION MASK FOR SECOND ERROR WORD
5289  012506  000000                ECWRD:  .WORD   0               ;LOCATION OF FIRST ERROR WORD
5290  012510  000000                ECGD:   .WORD   0               ;GOOD DATA, FIRST WORD
5291  012512  000000                ECBAD0: .WORD   0               ;BAD DATA, FIRST WORD
5292  012514  000000                ECWRD1: .WORD   0               ;LOCATION OF SECOND ERROR WORD
5293  012516  000000                ECGD1:  .WORD   0               ;GOOD DATA, SECOND WORD
5294  012520  000000                ECBAD1: .WORD   0               ;BAD DATA, ECOND WORD
5295
5296                                ;ROUTINE TO DISPLAY THE SECTOR WHICH GAVE THE HARD ERROR
5297
5298  012522  032737  000010  177570  PRTBAD: BIT     #SW3,SWR        ;PRINT THE BAD SECTOR ?
5299  012530  001460                        BEQ     6$              ;BR IF NOT
5300  012532  016001  000170                MOV     $RHBA(R0),R1    ;PUT THE END ADDRESS INTO R1
5301  012536  016046  000020                MOV     $WRDL(R0),-(SP) ;FIND THE BEGINNING OF THE SECTOR
5302  012542  066016  000166                ADD     $RHWC(R0),(SP)  ;SUBTRACT THE WORDS NOT TRANSFERED
5303  012546  005046                        CLR     -(SP)           ;MAKE THE UPPER DIVIDEND 0
5304  012550  016046  000026                MOV     $SSEC(R0),-(SP)  ;DIVDE THE WORDS TRANSFERED BY THE SECTOR SIZE
5305  012554  004737  024136                JSR     PC,LINKDV       ;DIVIDE
5306  012560  005716                        TST     (SP)            ;REMAINDER = 0 ?
5307  012562  001403                        BEQ     1$              ;BR IF IT IS - COMPLETE SECTOR TRANSFERED
5308  012564  006316                        ASL     (SP)            ;CONVERT THE RESIDUAL SECTOR SIZE INTO BYTE COUNT
5309  012566  161601                        SUB     (SP),R1         ;SUBTRACT IT FROM THE END ADDRESS
5310  012570  000410                        BR      2$              ;FINISH THE SIZING
5311  012572  162701  001000        1$:     SUB     #1000,R1        ;SUBTRACT FULL SECTOR SIZE FROM END ADDR
5312  012576  126027  000030  000005        CMPB    $CODE(R0),#5    ;WAS OPERATION READ HEADER & DATA ?
5313  012604  001002                        BNE     2$              ;BR IF NOT
5314  012606  162701  000010                SUB     #10,R1          ;SUBTRACK HEADER SIZE FROM ADDR
5315  012612  062706  000004        2$:     ADD     #4,SP           ;RESTORE THE STACK POINTER
5316  012616  104422  051177                DISPLY  ,LIN11H ;PRINT THE HEADER
5317  012622  012702  000007        3$:     MOV     #7,R2           ;R2 CONTAINS THE WORDS/LINE COUNT
5318  012626  010146                        MOV     R1,-(SP)        ;PUT THE ADDRESS ON THE STACK
5319  012630  004737  024652                JSR     PC,PRTOCT       ;PRINT IT
5320  012634  020160  000170        4$:     CMP     R1,$RHBA(R0)    ;PRINTED ALL THE SECTOR ?
5321  012640  001412                        BEQ     5$              ;BR IF ALL PRINTED
5322  012642  104422  051274                DISPLY  ,LINSP          ;SPACES
5323  012646  012146                        MOV     (R1)+,-(SP)     ;PUT THE DATA ON THE STACK
5324  012650  004737  024652                JSR     PC,PRTOCT       ;PRINT IT
5325  012654  005302                        DEC     R2              ;DECREMENT THE HORIZONTAL COUNT
5326  012656  001366                        BNE     4$              ;BR IF NOT AT THE END OF THE LINE
5327  012660  104422  001157                DISPLY  ,$CRLF          ;CR-LF
5328  012664  000756                        BR      3$              ;RESTORE THE WORDS/LINE COUNT
5329  012666  104422  001157        5$:     DISPLY  ,$CRLF          ;PRINT WHAT REMAINS IN THE BUFFER
5330  012672  000207                6$:     RTS     PC              ;RETURN
5331
5332                                ;ROUTINE TO DO AN RTC - UNIT SELECTED IN R0
5333
5334  012674  111037  041436        RTNCTR: MOVB    (R0),GENDPB     ;MOVE THE UNIT # TO THE GENERAL DPB
5335  012700  112737  000117  041440        MOVB    #RTC,GENDPB+$COMND  ;COMMAND CODE
5336  012706  004037  031352                JSR     R0,RP04         ;DRIVER ENTRANCE
5337  012712  041436                        GENDPB                  ;DPB ADDRESS FOR ORDER
5338  012714  000000                        HALT                    ;DRIVER DIDN'T ACCEPT ORDER
5339  012716  000207                        RTS     PC              ;RETURN
```

```
5340
5341                                    ;ROUTINE TO DO A RECALIBRATE - UNIT SELECTED IN R0
5342                                    ;       ENTER AT 'RECAL0' IF UNIT NUMBER ALREADY LOADED
5343                                    ;       INTO 'GENDPB'
5344
5345  012720  111037  041436  041440  RECALT: MOVB    (R0),GENDPB     ;MOVE THE UNIT # TO THE GENERAL DPB
5346  012724  112737  000107  041440  RECAL0: MOVB    #RECAL,GENDPB+$COMND  ;RELCALIBRATE COMMAND
5347  012732  004037  031352                  JSR     R0,RP04         ;DRIVER ENTRANCE
5348  012736  041436                          GENDPB                  ;DPB ADDRESS FOR ORDER
5349  012740  000000                          HALT                    ;DRIVER DIDN'T ACCEPT THE ORDER
5350  012742  005737  041454          1$:     TST     GENDPB+$STATUS          ;SEE IF FINISHED
5351  012746  001775                          BEQ     1$              ;BR IF NOT FINISHED
5352  012750  000207                          RTS     PC              ;RETURN
5353
5354                                    ;OFFSET THE UNIT IN R0 (OFFSET CODE PRELOADED)
5355
5356  012752  111037  041436          OFFST:  MOVB    (R0),GENDPB     ;UNIT # TO GENERAL DPB
5357  012756  112737  000115  041440          MOVB    #OFFSET,GENDPB+$COMND ;COMMAND
5358  012764  004037  031352                  JSR     R0,RP04         ;DRIVER ENTRANCE
5359  012770  041436                          GENDPB                  ;DPB ADDRESS FOR ORDER
5360  012772  000000                          HALT                    ;DRIVER DIDN'T ACCEPT ORDER
5361  012774  000207                          RTS     PC
5362
5363                                    ;UTILITY READ HEADER ROUTINE
5364                                    ;       ENTER WITH TRACK AND SECTOR ADDRS ON THE STACK
5365
5366  012776  116637  000002  041447  READHD: MOVB    2(SP),GENDPB+$TRK ;TRACK ADDRESS
5367  013004  116637  000004  041446          MOVB    4(SP),GENDPB+$SEC ;SECTOR ADDRESS
5368  013012  111037  041436                  MOVB    (R0),GENDPB     ;DRIVE NUMBER
5369  013016  016037  000222  041450          MOV     $RHCC(R0),GENDPB+$CYL ;CYLINDER ADDRESS
5370  013024  112737  000173  041440          MOVB    #ROHD,GENDPB+$COMND ;COMMAND
5371  013032  004037  031352                  JSR     R0,RP04         ;DRIVER ENTRANCE
5372  013036  041436                          GENDPB                  ;DPB ADDRESS FOR ORDER
5373  013040  000000                          HALT                    ;DRIVER DIDN'T ACCEPT COMMAND
5374  013042  005737  041454          1$:     TST     GENDPB+$STATUS  ;FINISHED?
5375  013046  001775                          BEQ     1$              ;BR IF NOT
5376  013050  012666  000002                  MOV     (SP)+,2(SP)     ;ADJUST STACK FOR RETURN
5377  013054  005726                          TST     (SP)+
5378  013056  000207                          RTS     PC              ;RETURN
5379
5380                                    ;RETRY THE PRESENT OPERATION
5381                                    ;       RETURN IF RETRY UNSUCESSFUL
5382                                    ;       RETURN+2 IF RETRY SUCESSFUL
5383                                    ;       RETURN TO MAIN PROGRAM IF DIFFERENT ERROR OCCURS
5384
5385  013060  004737  014252          RETRY:  JSR     PC,GODRIV       ;RE-START ORDER
5386  013064  005760  000016          1$:     TST     $STATUS(R0)     ;ORDER FINISHED?
5387  013070  001775                          BEQ     1$              ;BR IF NOT
5388  013072  100405                          BMI     2$              ;BR IF ERROR
5389  013074  105260  000022                  INCB    $RETRY(R0)      ;INCREMENT RETRY COUNT
5390  013100  062716  000002                  ADD     #2,(SP)         ;INCREMENT RETURN
5391  013104  000424                          BR      5$              ;GO TO EXIT
5392  013106  032760  000200  000016  2$:     BIT     #BIT7,$STATUS(R0)  ;DID ORDER TERMINATE NORMALLY ?
5393  013114  001427                          BEQ     7$              ;BR IF NOT
```

```
 5394  013116  005760  000024                    TST     $MASK(R0)          ;IS ERROR MASK 0 ?
 5395  013122  001004                            BNE     3$                 ;BR IF NOT
 5396  013124  005760  000200                    TST     $RHER1(R0)         ;MAKE SURE THAT THE DRIVE ERROR REG IS CLEAR
 5397  013130  001013                            BNE     6$                 ;BR IF NOT
 5398  013132  000404                            BR      4$                 ;CONTINUE RETRY
 5399  013134  036060  000024  000200  3$:      BIT     $MASK(R0),$RHER1(R0) ;SAME ERROR?
 5400  013142  001406                            BEQ     6$                 ;BR IF NOT
 5401  013144  105260  000022          4$:      INCB    $RETRY(R0)         ;INCREMENT RETRY COUNT
 5402  013150  105360  000023                    DECB    $RETRY+1(R0)       ;DECREMENT RETRY LIMIT
 5403  013154  001341                            BNE     RETRY              ;BR IF NOT DONE
 5404  013156  000207                    5$:     RTS     PC                 ;RETURN
 5405  013160  004737  020002          6$:     JSR     PC,LINE8           ;REPORT DIFFERENT ERROR
 5406  013164  004737  017534                    JSR     PC,LINE7           ;PRINT LINE 7
 5407  013170  005726                            TST     (SP)+              ;ADJUST STACK POINTER FOR DIRECT RETURN
 5408  013172  000207                            RTS     PC                 ;RETURN
 5409  013174  104422  050447          7$:     DISPLY  .LIN8M             ;'DIFFERENT ERROR DURING RETRY'
 5410  013200  000137  004472                    JMP     ERPRC1             ;REPORT THE ERROR
 5411
 5412                                    ;ROUTINE TO CLEAR THE DPB FOR THE ASSIGNED UNIT
 5413
 5414  013204                            CLRDPB:
   (2) 013204  010346                            MOV     R3,-(SP)           ;;PUSH R3 ON STACK
   (2) 013206  010446                            MOV     R4,-(SP)           ;;PUSH R4 ON STACK
   (2) 013210  010546                            MOV     R5,-(SP)           ;;PUSH R5 ON STACK
 5415  013212  010004                            MOV     R0,R4              ;GET THE DPB ADDRESS
 5416  013214  062704  000002                    ADD     #2,R4              ;ADDRESS OF FIRST LOCN TO BE CLEARED
 5417  013220  012703  000005                    MOV     #5,R3              ;NUMBER OF LOCNS TO BE CLEARED
 5418  013224  005024                            CLR     (R4)+              ;CLEAR THE LOCN
 5419  013226  005303                            DEC     R3                 ;DECREMENT THE LOCN COUNTER
 5420  013230  001375                            BNE     .-4                ;BR IF NOT FINISHED
 5421  013232  062704  000002                    ADD     #2,R4              ;MOVE THE ADDRESS PAST THE 'REG' ADDR
 5422  013236  012703  000220                    MOV     #$RHEC2+2-$REG,R3  ;NUMBER OF LOCNS TO BE CLEARED
 5423  013242  005024                            CLR     (R4)+              ;CLEAR
 5424  013244  162703  000002                    SUB     #2,R3              ;DECREMENT THE LOCN COUNTER
 5425  013250  001374                            BNE     .-6                ;BR IF NOT FINISHED
 5426  013252  012605                            MOV     (SP)+,R5           ;;POP STACK INTO R5
   (2) 013254  012604                            MOV     (SP)+,R4           ;;POP STACK INTO R4
   (2) 013256  012603                            MOV     (SP)+,R3           ;;POP STACK INTO R3
 5427  013260  000207                            RTS     PC
 5428
 5429                                    ;ROUTINE TO UPDATE THE GENERAL STATISTICS FOR THE DRIVE IN R0
 5430
 5431  013262  016037  000170  013414  STATIS: MOV     $RHBA(R0),FACTOR   ;STORE THE FINAL BUFFER ADDRESS
 5432  013270  166037  000006  013414          SUB     $BUF(R0),FACTOR    ;SUBTRACT THE INITIAL ADDRESS
 5433  013276  001434                            BEQ     2$                 ;BR IF NO DATA TRANSFER
 5434  013300  006237  013414                    ASR     FACTOR             ;CONVERT TO A WORD COUNT
 5435  013304  063760  013414  000052            ADD     FACTOR,$TRANS(R0)  ;UPDATE WORD COUNT
 5436  013312  005560  000054                    ADC     $TRANS+2(R0)
 5437  013316  132760  000002  000030            BITB    #BIT1,$CODE(R0)    ;SEE IF ORDER READ OR WRITE
 5438  013324  001021                            BNE     2$                 ;BRANCH IF ORDER WRITE
 5439  013326  005737  001310                    TST     AUTOCK             ;AUTO WRITE CHECKS BEING PERFORMED
 5440  013332  001411                            BEQ     1$                 ;BR IF NOT
 5441  013334  126027  000030  000001            CMPB    $CODE(R0),#1       ;PRESENT OPERATION AN AUTOMATIC WRITE CHECK ?
 5442  013342  101005                            BHI     1$                 ;BR IF NOT
```

```
5443  013344  066060  000020  000052          ADD     $WRDL(R0),$TRANS(R0)  ;ADD WORDS WRITTEN
5444  013352  005560  000054                  ADC     $TRANS+2(R0)          ;ADD A CARRY
5445  013356  063760  013414  000056  1$:     ADD     FACTOR,$READ(R0)      ;UPDATE THE READ WORD COUNT
5446  013364  005560  000060                  ADC     $READ+2(R0)
5447  013370  026060  000012  000220  2$:     CMP     $CYL(R0),$RHCA(R0)    ;DID MID-TRANSFER SEEK OCCUR
5448  013376  001405                          BEQ     3$                    ;BR IF NOT
5449  013400  062760  000001  000046          ADD     #1,$POSIT(R0)         ;INCREMENT SEEK COUNT
5450  013406  005560  000050                  ADC     $POSIT+2(R0)          ;ADD CARRY TO UPPER WORD
5451  013412  000207                  3$:     RTS     PC
5452
5453  013414  000000                  FACTOR: .WORD   0                     ;USED FOR WORDS TRANSFERED
5454
5455                                  ;ROUTINE TO GET A BUFFER
5456
5457  013416  010146                  GETBUF: MOV     R1,-(SP)              ;SAVE R1
5458  013420  022737  000002  041712          CMP     #2,BUFTBL             ;SEE HOW MANY BUFFER BLOCKS ARE IN TABLE
5459  013426  103001                          BHIS    .+4                   ;BR IF NOT MORE THAN 2
5460  013430  000240                          NOP                           ;CHANGE TO HALT FOR DEBUGGING
5461  013432  013702  041712                  MOV     BUFTBL,R2             ;NUMBER OF SEPARATE BUFFERS
5462  013436  001413                          BEQ     5$                    ;BR IF NONE AVAILABLE
5463  013440  012701  041714                  MOV     #BUFTBL+2,R1          ;FIRST ADDRESS OF ALLOCATION TABLE
5464  013444  026061  000020  000002  1$:     CMP     $WRDL(R0),2(R1)       ;SEE IF THERE IS A BLOCK LARGE ENOUGH
5465  013452  101407                          BLOS    3$                    ;BRANCH IF IT IS
5466  013454  005302                          DEC     R2                    ;DECREMENT TABLE COUNT
5467  013456  001403                          BEQ     5$                    ;BR IF THROUGH TABLE
5468  013460  062701  000004                  ADD     #4,R1                 ;INCREMENT TABLE POINTER
5469  013464  000767                          BR      1$                    ;CONTINUE LOOKING
5470  013466  012601                  5$:     MOV     (SP)+,R1              ;RESTORE R1
5471  013470  000207                          RTS     PC                    ;RETURN
5472  013472  011166  000004          3$:     MOV     (R1),4(SP)            ;BUFFER ADDRESS TO STACK
5473  013476  166061  000020  000002          SUB     $WRDL(R0),2(R1)       ;ADJUST BUFFER SIZE
5474  013504  001407                          BEQ     4$                    ;BR IF DIFFERENCE IS ZERO
5475  013506  006360  000020                  ASL     $WRDL(R0)             ;CONVERT # WORDS TO BYTES
5476  013512  066011  000020                  ADD     $WRDL(R0),(R1)        ;MAKE NEW STARTING ADDRESS
5477  013516  006260  000020                  ASR     $WRDL(R0)             ;RETURN # BYTES TO WORDS
5478  013522  000761                          BR      5$                    ;RETURN
5479  013524  005337  041712          4$:     DEC     BUFTBL                ;DECREMENT ENTRIES COUNT
5480  013530  001756                          BEQ     5$                    ;BR IF ALLOCATION TABLE EMPTY
5481  013532  005302                          DEC     R2                    ;DECREMENT TABLE COUNT
5482  013534  001754                          BEQ     5$                    ;BR IF ITEM WERE LAST ENTRY
5483  013536  010103                          MOV     R1,R3                 ;MOVE TABLE POINTER
5484  013540  062703  000004                  ADD     #4,R3                 ;POINT TO NEXT ENTRY
5485  013544  012321                  6$:     MOV     (R3)+,(R1)+           ;MOVE ITEMS
5486  013546  012321                          MOV     (R3)+,(R1)+
5487  013550  005302                          DEC     R2                    ;DECREMENT TABLE COUNT
5488  013552  001374                          BNE     6$                    ;CONTINUE IF NOT AT END OF TABLE
5489  013554  000744                          BR      5$                    ;RETURN
5490
5491
5492                                  ;ROUTINE TO PUT BUFFER BACK IN TABLE
5493
5494  013556  010146                  RELBUF: MOV     R1,-(SP)              ;SAVE R1
5495  013560  012701  041714                  MOV     #BUFTBL+2,R1          ;BEGINNING OF TABLE
5496  013564  013702  041712                  MOV     BUFTBL,R2             ;ENTRY COUNT
```

```
5497  013570  001424                        BEQ     2$              ;BR IF EMPTY TABLE
5498  013572  016003  000020                MOV     $WRDL(R0),R3    ;TRIAL ADDRESS
5499  013576  006303                        ASL     R3              ;CHANGE TO BYTE COUNT
5500  013600  066003  000006                ADD     $BUF(R0),R3     ;ADDRESS OF HIGHER ADJACENT BLOCK
5501  013604  021103              1$:       CMP     (R1),R3         ;UPPER ADJACENT BLOCK
5502  013606  001432                        BEQ     4$              ;BR IF YES
5503  013610  062701  000004                ADD     #4,R1           ;INCREMENT POINTER
5504  013614  005302                        DEC     R2              ;DECREMENT ENTRY COUNT
5505  013616  001372                        BNE     1$              ;CONTINUE SEARCHING
5506  013620  016011  000006                MOV     $BUF(R0),(R1)   ;PUT THE BUFFER BLOCK INTO THE TABLE
5507  013624  016061  000020  000002        MOV     $WRDL(R0),2(R1) ;BLOCK SIZE
5508  013632  005237  041712                INC     BUFTBL          ;INCREMENT ENTRY COUNT
5509  013636  005202                        INC     R2              ;INCREMENT R2 FOR USE LATER
5510  013640  000422                        BR      5$              ;SEE IF A LOWER ADJACENT BLOCK IS IN THE TABLE
5511  013642  016021  000006        2$:     MOV     $BUF(R0),(R1)+  ;BLOCK ADDRESS TO TABLE
5512  013646  016021  000020                MOV     $WRDL(R0),(R1)+ ;SIZE TO TABLE
5513  013652  005237  041712                INC     BUFTBL          ;INCREMENT ENTRY COUNT
5514  013656  022737  000002  041712  3$:    CMP     #2,BUFTBL       ;SEE HOW MANY BLOCKS ARE IN TABLE
5515  013664  103001                        BHIS    .+4             ;BR IF NOT MORE THAN 2
5516  013666  000240                        NOP                     ;DEBUGGING AID
5517  013670  012601                        MOV     (SP)+,R1        ;RESTORE R1
5518  013672  000207                        RTS     PC              ;RETURN
5519  013674  016011  000006        4$:     MOV     $BUF(R0),(R1)   ;RELEASED BUFFER IS LOWER ADJACENT
5520  013700  066061  000020  000002        ADD     $WRDL(R0),2(R1) ;INCREMENTED SIZE
5521  013706  010246              5$:       MOV     R2,-(SP)        ;SAVE R2
5522  013710  013702  041712                MOV     BUFTBL,R2       ;ENTRY COUNT
5523  013714  012705  041714                MOV     #BUFTBL+2,R5    ;BEGINNING OF TABLE
5524  013720  016504  000002        6$:     MOV     2(R5),R4        ;BLOCK SIZE (IN WORDS)
5525  013724  006304                        ASL     R4              ;CHANGE TO BYTE COUNT
5526  013726  061504                        ADD     (R5),R4         ;ADD BLOCK BEGINNING ADDRESS
5527  013730  020411                        CMP     R4,(R1)         ;R1 STILL POINTS TO INSERTED ENTRY
5528  013732  001406                        BEQ     8$              ;LOWER ADJACENT IN TABLE
5529  013734  062705  000004                ADD     #4,R5           ;INCREMENT POINTER
5530  013740  005302                        DEC     R2              ;DECREMENT ENTRY COUNT
5531  013742  001366                        BNE     6$              ;CONTINUE LOOKING
5532  013744  005726                        TST     (SP)+           ;RESTORE STACK POINTER
5533  013746  000743                        BR      3$              ;END
5534  013750  012602              8$:       MOV     (SP)+,R2        ;RESTORE R2
5535  013752  066165  000002  000002        ADD     2(R1),2(R5)     ;INCREMENT LOWER BLOCK LENGTH
5536  013760  005337  041712                DEC     BUFTBL          ;DECREMENT ENTRY COUNT
5537  013764  010105                        MOV     R1,R5           ;GET READY TO COMPRESS
5538  013766  062705  000004                ADD     #4,R5           ;INCREMENT TO NEXT ENTRY
5539  013772  012521              9$:       MOV     (R5)+,(R1)+     ;COMPRESS TABLE
5540  013774  012521                        MOV     (R5)+,(R1)+     ;MOVE SIZE FIELD DOWN
5541  013776  005302                        DEC     R2              ;DECREMENT ENTRY COUNT
5542  014000  001374                        BNE     9$              ;BR IF NOT FINISHED
5543  014002  000725                        BR      3$              ;FINISHED
5544
5545
5546                          ;FILL THE ASSIGNED BUFFER (IF WRITE OR WRITE CHECK ORDER)
5547
5548  014004  026027  000006  053564  FILBUF: CMP     $BUF(R0),#ENDPGM  ;MAKE SURE BUFFER IS ABOVE PROGRAM
5549  014012  103002                        BHIS    .+6             ;BR IF IT IS
5550  014014  000000                        HALT                    ;BUFFER ADDRESS WITHIN PROGRAM, ILLEGAL
```

```
5551  014016  000776                          BR      .-2             ;INTERLOCK THE HALT
5552  014020  023760  001272  000020          CMP     MAXDL,$WRDL(R0) ;SEE IF WORD LENGTH DOESN'T EXCEED MAXIMUM
5553  014026  103002                          BHIS    .+6             ;BR IF IT DOESN'T
5554  014030  000000                          HALT                    ;WORD LENGTH FOR OPERATION GREATER THAN AVAIL MEM
5555  014032  000776                          BR      .-2             ;INTERLOCK THE HALT
5556  014034  132760  000004  000030          BITB    #BIT02,$CODE(R0) ;SEE IF READ ORDER
5557  014042  001401                          BEQ     1$              ;BR IF WRITE
5558  014044  000207                          RTS     PC              ;RETURN
5559  014046  016001  000006          1$:     MOV     $BUF(R0),R1     ;BUFFER ADDRESS
5560  014052  016002  000020                  MOV     $WRDL(R0),R2    ;POSITIVE WORD COUNT
5561  014056  132760  000001  000030          BITB    #BIT00,$CODE(R0) ;SEE IF WRITE HEADER TYPE ORDER
5562  014064  001413                          BEQ     2$              ;BR IF NOT
5563  014066  016011  000012                  MOV     $CYL(R0),(R1)   ;CYLINDER ADDRESS
5564  014072  052721  010000                  BIS     #BIT12,(R1)+    ;SET FMT22 BIT
5565  014076  016021  000010                  MOV     $SEC(R0),(R1)+  ;MOVE SECTOR & TRACK
5566  014102  005021                          CLR     (R1)+           ;CLEAR FIRST KEY WORD
5567  014104  005021                          CLR     (R1)+           ;CLEAR THE SECOND
5568  014106  162702  000004                  SUB     #4,R2           ;ADJUST THE WORD COUNT
5569  014112  003422                          BLE     4$              ;BR IF END OF PATTERN
5570  014114  005004                  2$:     CLR     R4              ;CLEAR R4
5571  014116  116004  000034                  MOVB    $PATTC(R0),R4   ;RELATIVE PATTERN ADDRESS
5572  014122  001417                          BEQ     5$              ;BR IF RANDOM DATA
5573  014124  016405  042556                  MOV     STNDAT(R4),R5   ;PATTERN ADDRESS
5574  014130  012703  000020                  MOV     #20,R3          ;PATTERN COUNT
5575  014134  012521                  3$:     MOV     (R5)+,(R1)+     ;MOVE THE PATTERN INTO THE BUFFER
5576  014136  005302                          DEC     R2              ;DECREMENT THE WORD COUNT
5577  014140  001407                          BEQ     4$              ;BR IF DONE (WORD COUNT = 0)
5578  014142  005303                          DEC     R3              ;DECREMENT THE PATTERN COUNT
5579  014144  001373                          BNE     3$              ;BR IF MORE PATTERN
5580  014146  012703  000020                  MOV     #20,R3          ;RESTORE PATTERN COUNT
5581  014152  016405  042556                  MOV     STNDAT(R4),R5   ;RESTORE THE ADDRESS
5582  014156  000766                          BR      3$              ;CONTINUE DISTRIBUTING THE PATTERN
5583  014160  000207                  4$:     RTS     PC              ;RETURN
5584  014162  132760  000002  000030  5$:     BITB    #BIT01,$CODE(R0) ;WRITE CHECK ORDER ?
5585  014170  001007                          BNE     6$              ;BR IF NOT
5586  014172  016037  000076  027676          MOV     $RSAV(R0),$LONUM ;RESTORE THE RANDOM NUMBERS
5587  014200  016037  000100  027674          MOV     $RSAV+2(R0),$HINUM ;TO RECREATE THE WRITE CHECK BUFFER
5588  014206  000410                          BR      7$              ;FILL THE WRITE CHECK BUFFER
5589  014210  004737  027550          6$:     JSR     PC,$RAND        ;GET TWO RANDOM NUMBERS
5590  014214  013760  027676  000076          MOV     $LONUM,$RSAV(R0) ;SAVE THE RANDOM NUMBERS
5591  014222  013760  027674  000100          MOV     $HINUM,$RSAV+2(R0)
5592  014230  013721  027676          7$:     MOV     $LONUM,(R1)+    ;PUT FIRST WORD IN BUFFER
5593  014234  005302                          DEC     R2
5594  014236  001750                          BEQ     4$              ;BR IF DONE
5595  014240  013721  027674                  MOV     $HINUM,(R1)+    ;PUT SECOND WORD IN BUFFER
5596  014244  005302                          DEC     R2              ;DEC WORD COUNT
5597  014246  001744                          BEQ     4$              ;BR IF DONE
5598  014250  000757                          BR      6$
5599
5600                          ;START THE ORDER FOR THE DPB IN R0
5601
5602  014252  010046          GODRIV: MOV     R0,-(SP)        ;SAVE R0
5603  014254  010037  014264          MOV     R0,1$           ;CURRENT DPB ADDRESS
5604  014260  004037  031352          JSR     R0,RP04
```

```
5605  014264  000000                  1$:     0                       ;DPB ADDR GOES HERE
5606  014266  000000                          HALT                    ;DRIVER REJECTED REQUEST
5607  014270  012600                          MOV     (SP)+,R0        ;RESTORE R0
5608  014272  062760  000001  000042          ADD     #1,$OPERC(R0)   ;INCREMENT THE OPERATION COUNT
5609  014300  005560  000044                  ADC     $OPERC+2(R0)
5610  014304  026060  000040  000012          CMP     $PREVA+2(R0),$CYL(R0)  ;DID ORDER REQUIRE A CYLINDER CHANGE
5611  014312  001405                          BEQ     2$              ;BR IF NOT
5612  014314  062760  000001  000046          ADD     #1,$POSIT(R0)   ;INCREMENT SEEK COUNT
5613  014322  005560  000050                  ADC     $POSIT+2(R0)    ;ADD ANY CARRY
5614  014326  000207                  2$:     RTS     PC
5615
5616                                  ;GENERATE PARAMETERS FOR THE OPERATION
5617
5618  014330  004737  027550          SELPAR: JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
5619  014334  032737  000001  177570          BIT     #SWD,SWR        ;SEE IF SWD SET
5620  014342  001012                          BNE     2$              ;BR IF SET - READ ONLY
5621  014344  012705  000010          1$:     MOV     #10,R5          ;READ/WRITE SELECTION DIVISOR
5622  014350  004737  024110                  JSR     PC,GETREM       ;GET SELECTION VALUE
5623  014354  020537  001306                  CMP     R5,RATIO        ;DETERMINE IF READ OR WRITE
5624  014360  003003                          BGT     2$              ;BR IF READ
5625  014362  004737  014704                  JSR     PC,RANWRT       ;SELECT A WRITE ORDER
5626  014366  000406                          BR      3$              ;CONTINUE WITH THE SELECTION
5627  014370  013705  027676          2$:     MOV     $LONUM,R5       ;SELECT READ OPERATION CODE
5628  014374  042705  177776                  BIC     #^C1,R5         ;MASK OUT ALL BUT BIT 0
5629  014400  062705  000004                  ADD     #4,R5           ;TABLE OFFSET FOR READ CODE
5630  014404  110560  000102          3$:     MOVB    R5,$NCODE(R0)   ;ORDER SELECTION CODE TO CONTROL BLOCK
5631
5632                                  ;GENERATE A RANDOM SECTOR ADDRESS BETWEEN VALUES 'MINSEC' & 'MAXSEC'
5633
5634  014410  013705  001326                  MOV     MAXSEC,R5       ;GET MAXIMUM SECTOR ADDRESS
5635  014414  163705  001330                  SUB     MINSEC,R5       ;SUBTRACT MINIMUM SECTOR ADDRESS
5636  014420  001403                          BEQ     4$              ;BR IF MIN & MAX THE SAME
5637  014422  005205                          INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
5638  014424  004737  024110                  JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
5639  014430  063705  001330          4$:     ADD     MINSEC,R5       ;CONVERT TO ADDRESS
5640  014434  110560  000104                  MOVB    R5,$NSEC(R0)    ;STORE SECTOR ADDRESS IN DPB
5641
5642                                  ;GENERATE A RANDOM TRACK ADDRESS BETWEEN VALUES 'MINTRK' & 'MAXTRK'
5643
5644  014440  013705  001322                  MOV     MAXTRK,R5       ;GET MAXIMUM TRACK ADDRESS
5645  014444  163705  001324                  SUB     MINTRK,R5       ;SUBTRACT MINIMUM TRACK ADDRESS
5646  014450  001403                          BEQ     5$              ;BR IF MIN & MAX THE SAME
5647  014452  005205                          INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
5648  014454  004737  024110                  JSR     PC,GETREM       ;GET THE RANDOM TRACK AUGMENT
5649  014460  063705  001324          5$:     ADD     MINTRK,R5       ;CONVERT AUGMENT TO AN ADDRESS
5650  014464  110560  000105                  MOVB    R5,$NTRK(R0)    ;STORE TRACK ADDRESS IN DPB
5651
5652                                  ;GENERATE A RANDOM CYLINDER ADDRESS BETWEEN VALUES 'MINCYL' & 'MAXCYL'
5653
5654  014470  013705  001316                  MOV     MAXCYL,R5       ;GET MAXIMUM CYLINDER ADDRESS
5655  014474  163705  001320                  SUB     MINCYL,R5       ;SUBTRACT MINIMUM CYLINDER ADDRESS
5656  014500  001403                          BEQ     6$              ;BR IF MIN & MAX THE SAME
5657  014502  005205                          INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
5658  014504  004737  024110                  JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
```

```
5659  014510  063705  001320          6$:     ADD     MINCYL,R5       ;CONVERT AUGMENT TO AN ADDRESS
5660  014514  010560  000106                  MOV     R5,$NCYL(R0)    ;;STORE CYLINDER ADDRESS IN DPB
5661  014520  122760  000003  000102        CMPB    #3,$NCODE(R0)   ;WRITE HEADER & DATA ?
5662  014526  001444                        BEQ     10$             ;BR IF IT IS
5663
5664                                  ;GENERATE A RANDOM BUFFER LENGTH BETWEEN 4 & THE VALUE IN 'MAXDL'
5665
5666  014530  013705  001272          7$:     MOV     MAXDL,R5        ;GET BUFFER SIZE
5667  014534  005205                        INC     R5              ;INCREMENT THE MAXIMUM SIZE
5668  014536  004737  024110                JSR     PC,GETREM       ;DIVIDE BY MAX VALUE
5669  014542  005705                        TST     R5              ;IS THE REMAINDER 0 ?
5670  014544  001003                        BNE     8$              ;NOT 0, CONTINUE
5671  014546  004737  027550                JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
5672  014552  000766                        BR      7$              ;TRY AGAIN
5673  014554  010560  000110          8$:     MOV     R5,$NWRDL(R0)   ;WORD LENGTH TO CONTROL BLOCK
5674  014560  010546                        MOV     R5,-(SP)        ;NEW WORD LENGTH ON STACK FOR CHECK
5675  014562  005046                        CLR     -(SP)           ;MAKE UPPER DIVIDEND ZERO
5676  014564  012746  000400                MOV     #256.,-(SP)     ;SECTOR SIZE IS THE DIVISOR
5677  014570  132760  000001  000102        BITB    #1,$NCODE(R0)   ;SEE IF NEXT ORDER IS A HEADER ORDER
5678  014576  001402                        BEQ     .+6             ;BR IF NOT
5679  014600  062716  000004                ADD     #4,(SP)         ;ADD HEADER SIZE TO SECTOR SIZE
5680  014604  004737  024136                JSR     PC,LINKDV       ;DIVIDE BUFFER SIZE BY SECTOR SIZE
5681  014610  012616                        MOV     (SP)+,(SP)      ;MOV REMAINDER UP THE STACK
5682  014612  022627  000004                CMP     (SP)+,#4        ;SEE IF REMAINDER LESS THAN 4
5683  014616  103003                        BHIS    9$              ;BR IF NOT
5684  014620  004737  027550                JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
5685  014624  000741                        BR      7$              ;TRY AGAIN
5686  014626  122760  000002  000102  9$:     CMPB    #2,$NCODE(R0)   ;SEE IF WRITE DATA
5687  014634  001017                        BNE     12$             ;BR IF NOT WRITE DATA
5688  014636  000412                        BR      11$             ;GET PATTERN
5689
5690                                  ;SETUP BUFFER SIZE FOR A WRITE HEADER AND DATA ORDER
5691
5692  014640  012760  000404  000110  10$:     MOV     #260.,$NWRDL(R0)  ;CHANGE WORD LENGTH TO 260 FOR WRTHD ORDER
5693  014646  023727  001272  000404          CMP     MAXDL,#260.     ;CAN A FULL SECTOR BE WRITTEN ?
5694  014654  103003                        BHIS    11$             ;BR IF IT CAN
5695  014656  013760  001272  000110        MOV     MAXDL,$NWRDL(R0)  ;CHANGE TRANSFER SIZE
5696  014664  004737  015010          11$:    JSR     PC,GETPAT       ;GET PATTERN CODE
5697  014670  110560  000103                MOVB    R5,$NPATC(R0)   ;MOVE PATTERN CODE TO CONTROL BLOCK
5698  014674  012760  177777  000112  12$:     MOV     #-1,$NEXT(R0)  ;SET PARAMETERS SELECTED INDICATOR
5699  014702  000207                        RTS     PC              ;RETURN
5700
5701                                  ;ROUTINE TO SELECT A WRITE (OR WRITE CHECK) OPERATION
5702
5703  014704  012705  000004          RANWRT: MOV     #4,R5           ;WRITE OPERATION SELECTION DIVISOR
5704  014710  004737  024110                JSR     PC,GETREM       ;GET SELECTION CODE
5705  014714  005737  001310                TST     AUTOCK          ;ARE WRITE CHECK ORDERS TO BE SELECTED
5706                                                                ;RANDOMLY ?
5707  014720  001403                        BEQ     1$              ;BR IF THEY ARE
5708  014722  152705  000002                BISB    #2,R5           ;SET CODE TO EXCLUDE WRITE CHECK ORDERS
5709  014726  000420                        BR      3$              ;COMPLETE SELECTION
5710  014730  020527  000001          1$:     CMP     R5,#1           ;WRITE CHECK SELECTED ?
5711  014734  101015                        BHI     3$              ;BR IF NOT
5712  014736  132760  000002  000030        BITB    #2,$CODE(R0)    ;PREVIOUS WRITE OPERATION ?
```

```
5713  014744  001407                          BEQ     2$              ;BR IF PREVIOUS WAS READ OR WRITE CHECK
5714  014746  116060  000030  000102          MOVB    $CODE(R0),$NCODE(R0)  ;MOVE CODE TO 'NEXT CODE'
5715  014754  142760  000002  000102          BICB    #2,$NCODE(R0)   ;CHANGE WRITE TO WRITE CHECK
5716  014762  000411                          BR      5$              ;EXIT
5717  014764  052705  000002          2$:     BIS     #2,R5           ;CHANGE WRITE CHECK TO WRITE
5718  014770  005737  001302          3$:     TST     FORMAT          ;WRITE HEADER ORDERS ALLOWED ?
5719  014774  001002                          BNE     4$              ;BR IF THEY ARE
5720  014776  042705  000001                  BIC     #1,R5           ;ALTER POSSIBLE WRITE HEADER
5721  015002  110560  000102          4$:     MOVB    R5,$NCODE(R0)   ;SETUP 'NEXT' CODE
5722  015006  000207                  5$:     RTS     PC              ;RETURN
5723
5724                                  ;ROUTINE TO SELECT A PATTERN
5725
5726  015010  012705  000020          GETPAT: MOV     #20,R5          ;SELECT PATTERN
5727  015014  004737  024113                  JSR     PC,GETREM       ;GET CODE
5728  015020  005705                          TST     R5              ;WAS PATTERN ZERO SELECTED ?
5729  015022  001003                          BNE     1$              ;BR IF NOT ZERO
5730  015024  004737  027550                  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
5731  015030  000767                          BR      GETPAT          ;TRY AGAIN
5732  015032  006305                  1$:     ASL     R5              ;MAKE CODE INTO TABLE INDEX
5733  015034  000207                          RTS     PC
5734
5735                                  ;ROUTINE TO GET THE PREVIOUSLY SELECTED PARAMETER VALUES
5736
5737  015036  010546          GETPAR: MOV     R5,-(SP)        ;SAVE R5
5738  015040  116060  000030  000032          MOVB    $CODE(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
5739  015046  032760  000006  000102          BIT     #6,$NCODE(R0)   ;SEE IF NEXT OPERATION IS READ OR WRITE
5740  015054  001007                          BNE     1$              ;BR IF EITHER
5741  015056  016060  000012  000040          MOV     $CYL(R0),$PREVA+2(R0)  ;SAVE STARTING CYLINDER
5742  015064  016060  000010  000036          MOV     $SEC(R0),$PREVA(R0)  ;SAVE STARTING SECTOR AND TRACK
5743  015072  000405                          BR      2$
5744  015074  004737  015536          1$:     JSR     PC,PREVST       ;SAVE CURRENT SECTOR & TRACK
5745  015100  016060  000222  000040          MOV     $RHCC(R0),$PREVA+2(R0)  ;CURRENT CYLINDER
5746  015106  032737  000100  177570  2$:     BIT     #SW06,SWR       ;SWITCH 6 SET ?
5747  015114  001043                          BNE     3$              ;BR IF SET
5748  015116  116060  000102  000030          MOVB    $NCODE(R0),$CODE(R0)  ;LOGICAL CODE FOR OPERATION
5749  015124  116005  000102                  MOVB    $NCODE(R0),R5   ;LOAD R5 FOR USE AS TABLE INDEX
5750  015130  116560  042054  000002          MOVB    COMTBL(R5),$COMND(R0)  ;RP04 COMMAND CODE
5751  015136  116060  000103  000034          MOVB    $NPATC(R0),$PATTC(R0)  ;PATTERN CODE
5752  015144  016060  000104  000010          MOV     $NSEC(R0),$SEC(R0)  ;TRACK AND SECTOR ADDRESSES
5753  015152  016060  000106  000012          MOV     $NCYL(R0),$CYL(R0)  ;CYLINDER ADDRESS
5754  015160  016060  000110  000020          MOV     $NWRDL(R0),$WRDL(R0)  ;BUFFER SIZE
5755  015166  016060  000110  000004          MOV     $NWRDL(R0),$WRDM(R0)  ;WORD COUNT FOR THE RH11
5756  015174  005460  000004                  NEG     $WRDM(R0)       ;COMPLEMENT IT
5757  015200  012760  000400  000026          MOV     #256.,$SSEC(R0) ;INITIAL VALUE OF SECTOR SIZE
5758  015206  032760  000001  000030          BIT     #1,$CODE(R0)    ;HEADER OPERATION ?
5759  015214  001403                          BEQ     3$              ;BR IF NOT
5760  015216  062760  000004  000026          ADD     #4,$SSEC(R0)    ;ADD HEADER SIZE
5761  015224  005060  000112          3$:     CLR     $NEXT(R0)       ;RESET 'PARAMETERS LOADED' INDICATOR
5762  015230  012605                          MOV     (SP)+,R5        ;RESTORE R5
5763  015232  000207                          RTS     PC              ;RETURN
5764
5765                                  ;ROUTINE TO COMPRESS THE TABLE IN R1
5766
```

```
                                CMPRES: MOV     2(R1),(R1)      ;COMPRESS THE TABLE IN R1
                                        BEQ     1$              ;BR WHEN ZERO FOUND
                                        ADD     #2,R1           ;INCREMENT R1
                                        BR      CMPRES          ;CONTINUE COMPRESSING TABLE
                                1$:     RTS     PC              ;RETURN

                                ;ROUTINE TO SETUP PARAMETERS FOR A SEQUENTIAL READ OR WRITE OF THE DISK

                                WRTPK:  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
                                        TST     $OPERC+2(R0)    ;SEE IF FIRST OPERATION
                                        BNE     WRTPK1          ;BR IF UPPER WORD OF COUNTER NOT ZERO
                                        TST     $OPERC(R0)      ;LOWER WORD ZERO ?
                                        BNE     WRTPK1          ;BR IF NOT 1ST OPERATION
                                        TSTB    $PACK(R0)       ;SEE WHICH - 'R' OR 'W'
                                        BMI     WRTPK3          ;BR IF 'W'
                                        BR      WRTPK2          ;'R' OPERATION
                                WRTPK1: MOVB    $CODE(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
                                        JSR     PC,PREVST       ;SAVE CURRENT SECTOR & TRACK
                                        MOV     $RHCC(R0),$PREVA+2(R0)  ;CURRENT CYLINDER
                                        MOV     $RHDA(R0),$SEC(R0)  ;NEW SECTOR & TRACK ADDRESS
                                        MOV     $RHCA(R0),$CYL(R0)  ;NEW CYLINDER ADDRESS
                                        CMP     $CYL(R0),MAXCYL   ;SEE IF AT END
                                        BLO     2$              ;BR IF LESS THAN 'MAXCYL'
                                        BHI     1$              ;BR IF GREATER THAN 'MAXCYL'
                                        CMPB    $TRK(R0),MAXTRK   ;SEE IF AT MAX TRACK
                                        BLOS    2$              ;BR IF NOT GREATER
                                1$:     MOVB    MINTRK,$TRK(R0)   ;RESET TRACK ADDRESS
                                        MOVB    MINSEC,$SEC(R0)   ;RESET SECTOR ADDRESS
                                        MOV     MINCYL,$CYL(R0)   ;RESET CYLINDER ADDRESS
                                        JSR     PC,NRML2        ;DROP THE UNIT (NORMAL TERMINATION)
                                        BIT     #SW04,SWR       ;IS SWITCH 4 SET ?
                                        BNE     2$              ;BR IF SET
                                        TST     (SP)+           ;INCREMENT THE STACK POINTER
                                        JMP     MAIN            ;RETURN DIRECTLY TO 'MAIN'
                                2$:     MOV     MAXDL,$WRDL(R0)   ;BUFFER SIZE IS MAXIMUM
                                        MOV     MAXDL,$WRDM(R0)   ;WORD COUNT
                                        NEG     $WRDM(R0)       ;CHANGE WORD COUNT TO 2'S COMPLEMENT
                                        TSTB    $PACK(R0)       ;READ OR WRITE ?
                                        BMI     WRTPK3          ;BR IF WRITE
                                WRTPK2: MOV     #260.,$SSEC(R0)   ;SECTOR SIZE FOR READ
                                        MOVB    #5,$CODE(R0)    ;CODE FOR READ HEADER & DATA
                                        MOVB    #RDHD,$COMND(R0)  ;DRIVE CODE FOR OPERATION
                                        BR      WRTPK4          ;SET UP FOR EXIT
                                WRTPK3: MOV     #256.,$SSEC(R0) ;SECTOR SIZE
                                        MOVB    #2,$CODE(R0)    ;CODE FOR WRTDAT
                                        MOVB    #WRTDAT,$COMND(R0)  ;OP CODE
                                        JSR     PC,GETPAT       ;GET PATTERN CODE
                                        MOVB    R5,$PATTC(R0)   ;PATTERN CODE
                                WRTPK4: CLR     $NEXT(R0)       ;CLEAR 'PARAMETER SELECTED' INDICATOR
                                        RTS     PC              ;RETURN

                                ;DECREMENT AND SAVE CURRENT SECTOR & TRACK ADDRESS

                                PREVST: CLR     -(SP)           ;MAKE ROOM ON THE STACK
```

```
 5821  015540  005046                        CLR     -(SP)           ;MAKE ROOM ON THE STACK
 5822  015542  004737  020036                JSR     PC,READDR       ;DECREMENT SECTOR-TRACK ADDRESS
 5823  015546  112660  000037                MOVB    (SP)+,$PREVA+1(R0)  ;SAVE CURRENT TRACK
 5824  015552  112660  000036                MOVB    (SP)+,$PREVA(R0)  ;SAVE CURRENT SECTOR
 5825  015556  000207                        RTS     PC              ;RETURN
 5826
 5827
 5828                                  ;ROUTINE TO DETERMINE OF ERROR IS AT A LOCATION ON THE PACK DEFINED
 5829                                  ;       IN THE BAD TRACK/SECTOR TABLE FOR THE DRIVE.
 5830                                  ;       RETURN IF ERROR IS AT AN ADDRESS WHICH IS IN THE TABLE
 5831                                  ;       RETURN+2 IS THE ERROR ADDRESS IS NOT CONTAINED IN THE TABLE
 5832                                  ;       OR IF THE PARAMETER 'NOTPRT' IS 0
 5833
 5834  015560                          SPOTCK:
  (2)  015560  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  015562  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 5835  015564  012701  000114                MOV     #$BDSEC,R1      ;INCREMENT FOR BAD SECTOR TABLE
 5836  015570  060001                        ADD     R0,R1           ;ADD THE BLOCK'S STARTING ADDRESS
 5837  015572  012702  000010                MOV     #8.,R2          ;BAD SECTOR TABLE SIZE COUNT
 5838  015576  021160  000222          1$:   CMP     (R1),$RHCC(R0)  ;IS CYLINDER IN THE TABLE ?
 5839  015602  001021                        BNE     4$              ;BR IF NOT
 5840  015604  005046                        CLR     -(SP)           ;MAKE ROOM ON THE STACK
 5841  015606  005046                        CLR     -(SP)           ;MAKE ROOM ON THE STACK
 5842  015610  004737  020036                JSR     PC,READDR       ;DECREMENT THE SECTOR/TRACK ADDRESS
 5843  015614  122661  000003                CMPB    (SP)+,3(R1)     ;COMPARE THE TRACK ADDRESS
 5844  015620  001011                        BNE     3$              ;BR IF IT IS NOT EQUAL
 5845  015622  105761  000002                TSTB    2(R1)           ;IS A SECTOR ADDRESS IN THE TABLE ?
 5846  015626  100002                        BPL     2$              ;BR IF ONE IS
 5847  015630  005726                        TST     (SP)+           ;INCREMENT THE STACK POINTER
 5848  015632  000414                        BR      5$              ;DISPLAY THE MESSAGE
 5849  015634  122661  000002          2$:   CMPB    (SP)+,2(R1)     ;COMPARE THE SECTOR ADDRESS
 5850  015640  001002                        BNE     4$              ;BR IF NOT EQUAL
 5851  015642  000410                        BR      5$              ;CHECK 'NOTPRT'
 5852  015644  005726                  3$:   TST     (SP)+           ;INCREMENT THE STACK POINTER
 5853  015646  062701  000004          4$:   ADD     #4,R1           ;GO TO THE NEXT LOCATION IN THE TABLE
 5854  015652  005711                        TST     (R1)            ;PAST THE TABLE ENTRIES ?
 5855  015654  100411                        BMI     6$              ;BR IF PAST
 5856  015656  005302                        DEC     R2              ;DECREMENT THE MAXIMUM ENTRY COUNT
 5857  015660  001346                        BNE     1$              ;BR IF MORE TO CHECK
 5858  015662  000406                        BR      6$              ;END, EXIT
 5859  015664  005737  001312          5$:   TST     NOTPRT          ;PRINT THE ERROR ANYWAY ?
 5860  015670  001006                        BNE     7$              ;BR IF NOT
 5861  015672  012737  177777  016330        MOV     #-1,PRT2B       ;SET THE INDICATOR FOR THE IDENTIFICATION LINE
 5862  015700  062766  000002  000004  6$:   ADD     #2,4(SP)        ;INCREMENT THE RETURN
 5863  015706                          7$:
  (2)  015706  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  015710  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 5864  015712  000207                        RTS     PC              ;RETURN
 5865
 5866                                  ;******************************************************************
 5867
 5868                                  .SBTTL  ERROR MESSAGE GENERATION ROUTINES
 5869
 5870                                  ;******************************************************************
```

```
5871
5872                                        ;PRINT LINE 1 OF ERROR MESSAGE
5873
5874  015714  032737  002000  177570  LINE1:  BIT     #SW10,SWR       ;SWITCH 10 SET ?
5875  015722  001402                          BEQ     .+6             ;BR IF NOT
5876  015724  104400  001152                  TYPE    ,$BELL          ;RING THE BELL
5877  015730  104422  001157                  DISPLY  ,$CRLF          ;CR-LF
5878  015734  005737  001202                  TST     CLKFLG          ;CLOCK ON THE SYSTEM ?
5879  015740  00.006                          BNE     1$              ;BR IF NONE IS
5880  015742  104422  021614                  DISPLY  ,HOUR           ;TYPE THE HOURS
5881  015746  104422  021620                  DISPLY  ,MINUTE         ;TYPE THE MINUTES
5882  015752  104422  021624                  DISPLY  ,SECOND         ;TYPE THE SECONDS
5883  015756  000207                  1$:     RTS     PC              ;RETURN & TYPE DESCRIPTION
5884
5885                                        ;PRINT LINE 2 OF ERROR MESSAGE
5886
5887  015760  104422  001157          LINE2:  DISPLY  ,$CRLF          ;CR-LF
5888  015764  104422  047300                  DISPLY  ,LIN2C          ;'PRESENT ORDER = '
5889  015770  005037  016100                  CLR     2$              ;CLEAR LOCN FOR ORDER CODE
5890  015774  116037  000030  016100          MOVB    $CODE(R0),2$    ;ORDER CODE
5891  016002  004737  016054                  JSR     PC,1$           ;PRINT THE MNEMONIC OF THE ORDER
5892  016006  104422  047321                  DISPLY  ,LIN2P          ;'PREVIOUS ORDER = '
5893  016012  005037  016100                  CLR     2$              ;CLEAR FOR PREVIOUS ORDER CODE
5894  016016  116037  000032  016100          MOVB    $PREVO(R0),2$   ;PREVIOUS ORDER CODE
5895  016024  004737  016054                  JSR     PC,1$           ;PRINT THE MNEMONIC
5896  016030  032760  000040  000016          BIT     #BIT5,$TATUS(R0)  ;DID ERROR OCCUR DURING NON-DATA TRANSFER
5897  016036  001422                          BEQ     LINE2A          ;BR IF NOT
5898  016040  104422  001157                  DISPLY  ,$CRLF          ;CR-LF
5899  016044  104422  047345                  DISPLY  ,LIN2M          ;'ERROR OCCURED DURING NON-DATA TRANSFER'
5900  016050  000137  016104                  JMP     LINE2A          ;PRINT THE REST OF THE ERROR LINE
5901  016054  006337  016100          1$:     ASL     2$              ;CONVERT CODE INTO A TABLE INDEX
5902  016060  006337  016100                  ASL     2$              ;CONVERT CODE INTO A TABLE INDEX
5903  016064  006337  016100                  ASL     2$              ;CONVERT CODE INTO A TABLE INDEX
5904  016070  062737  042062  016100          ADD     #OPMNEM,2$      ;ADD THE TABLE BASE ADDRESS
5905  016076  104422                          DISPLY                  ;PRINT THE MNEMONIC
5906  016100  000000                  2$:     .WORD   0
5907  016102  000207                          RTS     PC              ;RETURN
5908  016104  005737  016330          LINE2A: TST     PRT2B           ;PRINT THE BAD SECTOR LINE ?
5909  016110  001404                          BEQ     LINE2B          ;BR IF NOT
5910  016112  104422  001157                  DISPLY  ,$CRLF          ;CR-LF
5911  016116  104422  047426                  DISPLY  ,LIN2S          ;ERROR ADDRESS DEFINED AS BAD AREA
5912  016122  010546                  LINE2B: MOV     R5 -(SP)        ;SAVE R5
5913  016124  005005                          CLR     R5              ;CLEAR R5 FOR DRIVE ADDRESS
5914  016126  111005                          MOVB    (R0),R5         ;MOVE DRIVE# FROM CURRENT BLOCK
5915  016130  142737  000007  001255          BICB    #7,UNIT+1       ;CLEAR UNIT NUMBER
5916  016136  150537  001255                  BISB    R5,UNIT+1       ;LOAD UNIT NUMBER
5917  016142  006305                          ASL     R5              ;MAKE DRIVE NUMBER INTO A TABLE INDEX
5918  016144  016537  042476  016322          MOV     DT14.T(R5),$LINAD  ;ADDRESS OF LOCATIONS TO PRINT
5919  016152  016537  042516  016324          MOV     DT15.T(R5),$LINAD+2  ;OPTIONAL REGISTERS
5920  016160  016537  042536  016326          MOV     DT16.T(R5),$LINAD+4  ;MORE OPTIONAL REGISTERS
5921  016166  104422  001157                  DISPLY  ,$CRLF          ;CR-LF
5922  016172  104422  046145                  DISPLY  ,DH14           ;STANDARD RP04 REGISTER HEADER
5923  016176  104422  001254                  DISPLY  ,UNIT           ;PRINT THE UNIT ADDRESS
5924  016202  104422  051274                  DISPLY  ,LINSP          ;SPACES
```

```
5925  016206  004737  016270                  JSR     PC,2$           ;PRINT THE REGISTERS
5926  016212  032737  000040  177570          BIT     #SW05,SWR       ;PRINT THE OPTIONAL REGISTERS ?
5927  016220  001021                          BNE     1$              ;BR IF NOT
5928  016222  104422  046251                  DISPLY  ,DH15
5929  016226  013737  016324  016322          MOV     $LINAD+2,$LINAD  ;SHIFT ADDRESSES
5930  016234  013737  016326  016324          MOV     $LINAD+4,$LINAD+2  ;FINISH SHIFT
5931  016242  004737  016270                  JSR     PC,2$           ;PRINT THEM
5932  016246  104422  046350                  DISPLY  ,DH16
5933  016252  013737  016324  016322          MOV     $LINAD+2,$LINAD  ;FINISH SHIFTING THE ADDRESSES
5934  016260  004737  016270                  JSR     PC,2$           ;PRINT THE REGISTERS
5935  016264  012605                  1$:     MOV     (SP)+,R5        ;RESTORE R5
5936  016266  000207                          RTS     PC
5937  016270  013705  016322          2$:     MOV     $LINAD,R5       ;ADDRESS OF REGISTERS
5938  016274  005715                  3$:     TST     (R5)            ;AT END OF LINE ?
5939  016276  001406                          BEQ     4$              ;BR IF YES
5940  016300  013546                          MOV     @(R5)+,-(SP)    ;PUT THE CONTENTS ON THE STACK
5941  016302  004737  024652                  JSR     PC,PRTOCT       ;PRINT THE NUMBER
5942  016306  104422  051274                  DISPLY  ,LINSP          ;PRINT 2 SPACES
5943  016312  000770                          BR      3$              ;CONTINUE PRINTING
5944  016314  104422  001157          4$:     DISPLY  ,$CRLF          ;CR-LF
5945  016320  000207                          RTS     PC
5946
5947  016322  000000  000000  000000  $LINAD: .WORD   0,0,0           ;ADDRESS OF REGISTERS
5948  016330  000000                  PRT28:  .WORD   0               ;BAD SECTOR LINE INDICATOR
5949
5950                                  ;PRINT LINES 3 & 3A OF ERROR MESSAGE
5951
5952  016332  104422  047462          LINE3:  DISPLY  ,LINM3          ;LINE 3 ENTRANCE
5953  016336  000402                          BR      LIN3.1          ;FINISH PRINTOUT
5954  016340  104422  047500          LINE3A: DISPLY  ,LINM3          ;LINE 3A ENTRANCE
5955  016344  016046  000222          LIN3.1: MOV     $RHCC(R0),-(SP) ;PUT CYLINDER ADDR ON STACK
5956  016350  004737  030170                  JSR     PC,$SB2D        ;CONVERT TO DECIMAL
5957  016354  004737  025510                  JSR     PC,$SUPRS       ;PRINT CYL ADDR
5958  016360  104422  047475                  DISPLY  ,T              ;PRINT ' T'
5959  016364  005046                          CLR     -(SP)           ;CLEAR STACK FOR SECTOR ADDR
5960  016366  005046                          CLR     -(SP)           ;CLEAR STACK FOR TRACK ADDR
5961  016370  004737  020036                  JSR     PC,READDR       ;SET DECREMENTED ADDRESS
5962  016374  004737  030170                  JSR     PC,$SB2D        ;NOW CONVERT TRACK
5963  016400  004737  025510                  JSR     PC,$SUPRS       ;PRINT TRACK VALUE
5964  016404  004737  030170                  JSR     PC,$SB2D        ;CONVERT SECTOR
5965  016410  104422  047521                  DISPLY  ,S              ;PRINT ' S'
5966  016414  004737  025510                  JSR     PC,$SUPRS       ;PRINT SECTOR
5967  016420  104422  047524                  DISPLY  ,LINP3          ;PRINT 'PREV ADDR'
5968  016424  016046  000040                  MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
5969  016430  004737  030170                  JSR     PC,$SB2D        ;CONVERT CYLINDER
5970  016434  004737  025510                  JSR     PC,$SUPRS       ;PRINT CYLINDER
5971  016440  104422  047475                  DISPLY  ,T              ;PRINT ' T'
5972  016444  005046                          CLR     -(SP)           ;MAKE ROOM ON THE STACK
5973  016446  116016  000037                  MOVB    $PREVA+1(R0),(SP)  ;PREVIOUS TRACK ADDRESS
5974  016452  004737  030170                  JSR     PC,$SB2D        ;CONVERT TRACK
5975  016456  004737  025510                  JSR     PC,$SUPRS       ;PRINT TRACK
5976  016462  104422  047521                  DISPLY  ,S              ;PRINT ' S'
5977  016466  005046                          CLR     -(SP)           ;MAKE ROOM ON THE STACK
5978  016470  116016  000036                  MOVB    $PREVA(R0),(SP)  ;PREVIOUS SECTOR DDRESS
```

```
                                        JSR     PC,$S820        ;CONVERT SECTOR
                                        JSR     PC,$SUPRS       ;PRINT SECTOR
                                        DISPLY  ,$CRLF
                                        RTS     PC

                                ;PRINT LINES 3A & 3C OF ERROR MESSAGE

                        LINE3B: JSR     PC,LIN3.3       ;LINE 3B ENTRANCE
                                DISPLY  ,$CRLF
                                RTS     PC
                        LINE3C: JSR     PC,LIN3.3       ;LINE 3C ENTRANCE
                                JMP     LIN3.4          ;FINISH MESSAGE

                        LIN3.3: DISPLY  ,LINS3          ;LINE '3B & 3C' ENTRANCE
                                MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
                                JSR     PC,LIN3.2       ;CONVERT & TYPE
                                DISPLY  ,LINEN3         ;PRINT 'END CYL'
                                MOV     $RHCC(R0),-(SP) ;PRESENT CYLINDER
                                JSR     PC,LIN3.2       ;CONVERT & TYPE
                                RTS     PC

                        LIN3.4: DISPLY  ,LINA3          ;PRINT 'ACTUAL'
                                MOV     CYLDER,-(SP)    ;ACTUAL CYLINDER
                                JSR     PC,LIN3.2       ;CONVERT & TYPE
                                DISPLY  ,LINT3          ;PRINT TRACK
                                CLR     -(SP)           ;CLEAR STACK WORD
                                MOVB    $RHDA+1(R0),(SP) ;PUT TRACK ON STACK
                                JSR     PC,LIN3.2       ;CONVERT & TYPE
                                DISPLY  ,$CRLF
                                RTS     PC

                        LIN3.2: MOV     2(SP),-(SP)     ;SET UP STACK FOR CONVERT
                                JSR     PC,$S820        ;CONVERT
                                JSR     PC,$SUPRS       ;TYPE
                                MOV     (SP)+,(SP)      ;RESTORE STACK POINTER
                                RTS     PC

                                ;PRINT LINE 3D OF ERROR MESSAGE

177570          LINE3D: BIT     #SW05,SWR       ;SWITCH 5 SET ?
                        BEQ     1$              ;BR IF IT IS
                        DISPLY  ,LIN83          ;'RHBA = '
                        MOV     $RHBA(R0),-(SP) ;BUFFER ADDR REG CONTENTS
                        JSR     PC,LIN4.1       ;CONVERT & PRINT IT
                        DISPLY  ,LINW3          ;'   RHWC = '
                        MOV     $RHWC(R0),-(SP) ;WORD COUNT REGISTER CONTENTS
                        JSR     PC,LIN4.1       ;CONVERT IT & PRINT IT
                        DISPLY  ,$CRLF
                1$:     RTS     PC

                                ;PRINT LINE 3E OF ERROR MESSAGE

                        LINE3E: DISPLY  ,LINS3          ;'START CYL = '
                                MOV     $CYL(R0),-(SP)  ;MOVE CYL TO STACK
```

```
6033  016724  004737  030170                  JSR     PC,$SB2D        ;CONVERT IT
6034  016730  004737  025510                  JSR     PC,$SUPRS       ;PRINT IT
6035  016734  104422  051274                  DISPLY  ,LINSP          ;SPACES
6036  016740  104422  047673                  DISPLY  ,LINST3         ;'START TRK = '
6037  016744  005046                          CLR     -(SP)           ;CLEAR STACK
6038  016746  116016  000011                  MOVB    $TRK(R0),(SP)   ;TRACK TO STACK
6039  016752  004737  030170                  JSR     PC,$SB2D        ;CONVERT IT
6040  016756  004737  025510                  JSR     PC,$SUPRS       ;PRINT IT
6041  016762  104422  051274                  DISPLY  ,LINSP          ;SPACES
6042  016766  104422  047710                  DISPLY  ,LINSS3         ;'START SEC = '
6043  016772  005046                          CLR     -(SP)           ;CLEAR STACK
6044  016774  116016  000010                  MOVB    $SEC(R0),(SP)   ;SECTOR ADDR TO STACK
6045  017000  004737  030170                  JSR     PC,$SB2D        ;CONVERT IT
6046  017004  004737  025510                  JSR     PC,$SUPRS       ;PRINT IT
6047  017010  104422  001157                  DISPLY  ,$CRLF
6048  017014  000207                          RTS     PC
6049
6050                                  ;PRINT LINE 3F OF ERROR MESSAGE
6051
6052  017016  032737  000040  177570  LINE3F: BIT     #SW5,SWR        ;SWITCH 5 SET ?
6053  017024  001420                          BEQ     1$              ;BR IF NOT
6054  017026  104422  047641                  DISPLY  ,LINDA3         ;'RHDA = '
6055  017032  016046  000172                  MOV     $RHDA(R0),-(SP) ;DESIRED ADDRESS ON STACK
6056  017036  004737  024652                  JSR     PC,PRTOCT       ;PRINT IT
6057  017042  104422  051274                  DISPLY  ,LINSP          ;SPACES
6058  017046  104422  047630                  DISPLY  ,LINCA3         ;' RHCA = '
6059  017052  016046  000220                  MOV     $RHCA(R0),-(SP) ;DESIRED CYLINDER ADDRESS
6060  017056  004737  024652                  JSR     PC,PRTOCT       ;PRINT IT
6061  017062  104422  001157                  DISPLY  ,$CRLF
6062  017066  000207                  1$:     RTS     PC
6063
6064                                  ;PRINT LINE 4 OF ERROR MESSAGE
6065
6066  017070  104422  047725          LINE4:  DISPLY  ,LIN4           ;'PRINT BUFFER'
6067  017074  016046  000006                  MOV     $BUF(R0),-(SP)  ;BUFFER ADDR ON STACK
6068  017100  004737  017150                  JSR     PC,LIN4.1       ;CONVERT TO OCTAL & PRINT
6069  017104  104422  047744                  DISPLY  ,LINS4          ;PRINT 'SIZE'
6070  017110  016046  000020                  MOV     $WRDL(R0),-(SP) ;BUFFER SIZE
6071  017114  004737  016626                  JSR     PC,LIN3.2       ;CONVERT TO DEC & PRINT
6072  017120  104422  047756                  DISPLY  ,LINX4          ;'ACTUAL NMBR WRDS XFRD = '
6073  017124  016046  000170                  MOV     $RHBA(R0),-(SP) ;VALUE IN BUFFER ADDR REGISTER
6074  017130  166016  000006                  SUB     $BUF(R0),(SP)   ;SUBTRACT STARTING ADDRESS
6075  017134  006216                          ASR     (SP)            ;CONVERT INTO A WORD COUNT
6076  017136  004737  016626                  JSR     PC,LIN3.2       ;CONVERT TO DEC & PRINT
6077  017142  104422  001157                  DISPLY  ,$CRLF
6078  017146  000207                          RTS     PC
6079
6080  017150  016646  000002          LIN4.1: MOV     2(SP),-(SP)     ;PUT VALUE IN PROPER LOCN ON STACK
6081  017154  004737  030340                  JSR     PC,$SB2O        ;CONVERT TO OCTAL
6082  017160  004737  025510                  JSR     PC,$SUPRS       ;PRINT IT
6083  017164  012616                          MOV     (SP)+,(SP)      ;RESTORE STACK POINTER
6084  017166  000207                          RTS     PC
6085
6086                                  ;PRINT LINE 5 OF ERROR MESSAGE
```

```
 6087
 6088  017170  104422  050011          LINES:  DISPLY  ,LIN5           ;PRINT 'GOOD DATA'
 6089  017174  162760  000002  000170          SUB     #2,$RHBA(R0)     ;BACK THE ADDRESS UP
 6090  017202  017046  000170                  MOV     @$RHBA(R0),-(SP)        ;'GOOD' DATA - AT THE RHBA LOCATION
 6091  017206  004737  024652                  JSR     PC,PRTOCT        ;PRINT IT
 6092  017212  104422  050026                  DISPLY  ,LIN8S           ;PRINT 'BAD DATA'
 6093  017216  016046  000206                  MOV     $RHDB(R0),-(SP)  ;GET BAD CHAR FROM BUFFER
 6094  017222  004737  024652                  JSR     PC,PRTOCT        ;PRINT IT
 6095  017226  016046  000166                  MOV     $RHWC(R0),-(SP)  ;WORD LENGTH ON STACK
 6096  017232  066016  000020                  ADD     $WRDL(R0),(SP)   ;MAKE INTO A POSITIVE NUMBER
 6097  017236  005046                          CLR     -(SP)            ;UPPER DIVIDEND TO ZERO
 6098  017240  016046  000026                  MOV     $SSEC(R0),-(SP)  ;SECTOR SIZE ON THE STACK
 6099  017244  004737  024136                  JSR     PC,LINKDV        ;DIVIDE WORDS XFERED BY SECTOR SIZE
 6100  017250  012616                          MOV     (SP)+,(SP)       ;MOVE REMAINDER UP THE STACK
 6101  017252  104422  050044                  DISPLY  ,LINP5           ;PRINT 'SECT POS'
 6102  017256  004737  016626                  JSR     PC,LIN3.2        ;CONVERT & PRINT IT
 6103  017262  104422  001157                  DISPLY  ,$CRLF
 6104  017266  000207                          RTS     PC
 6105
 6106                                  ;PRINT LINE 5A OF THE ERROR MESSAGE
 6107
 6108  017270  104422  050062          LINE5A: DISPLY  ,LIN5S           ;'HEADER CONTENTS OF ERROR SECTOR'
 6113  017274  013746  053554                  MOV     CYLDER,-(SP)     ;HEADER POSITION TO STACK
  (1)  017300  004737  024652                  JSR     PC,PRTOCT        ;PRINT HEADER POSITION+
  (1)  017304  104422  051274                  DISPLY  ,LINSP           ;SPACES
  (1)  017310  013746  053556                  MOV     CYLDER+2,-(SP)   ;HEADER POSITION+2 TO STACK
  (1)  017314  004737  024652                  JSR     PC,PRTOCT        ;PRINT HEADER POSITION++2
  (1)  017320  104422  051274                  DISPLY  ,LINSP           ;SPACES
  (1)  017324  013746  053560                  MOV     CYLDER+4,-(SP)   ;HEADER POSITION+4 TO STACK
  (1)  017330  004737  024652                  JSR     PC,PRTOCT        ;PRINT HEADER POSITION++4
  (1)  017334  104422  051274                  DISPLY  ,LINSP           ;SPACES
  (1)  017340  013746  053562                  MOV     CYLDER+6,-(SP)   ;HEADER POSITION+6 TO STACK
  (1)  017344  004737  024652                  JSR     PC,PRTOCT        ;PRINT HEADER POSITION++6
  (1)  017350  104422  051274                  DISPLY  ,LINSP           ;SPACES
 6114  017354  104422  001157                  DISPLY  ,$CRLF
 6115  017360  000207                          RTS     PC
 6116
 6117                                  ;PRINT LINE 5B OF ERROR MESSAGE
 6118
 6119  017362  104422  050116          LINE5B: DISPLY  ,LINEP5          ;'RHEC1 = '
 6120  017366  016046  000230                  MOV     $RHEC1(R0),-(SP) ;ECC POSITION REGISTER ON STACK
 6121  017372  004737  024652                  JSR     PC,PRTOCT        ;PRINT IT
 6122  017376  104422  051274                  DISPLY  ,LINSP           ;SPACES
 6123  017402  104422  050127                  DISPLY  ,LINE05          ;' RHEC2 = '
 6124  017406  016046  000232                  MOV     $RHEC2(R0),-(SP) ;ECC PATTERN REGISTER ON STACK
 6125  017412  004737  024652                  JSR     PC,PRTOCT        ;PRINT IT
 6126  017416  104422  001157                  DISPLY  ,$CRLF
 6127  017422  000207                          RTS     PC               ;RETURN
 6128
 6129                                  ;PRINT LINE 6 OF ERROR MESSAGE
 6130
 6131  017424  104422  050141          LINE6:  DISPLY  ,LIN86           ;ECC CORRECTABLE
 6132  017430  104422  001157                  DISPLY  ,$CRLF
 6133  017434  000207                          RTS     PC
```

```
                                        ;PRINT LINE 6A OF THE ERROR MESSAGE

017436  104422  050174        LINE6A: DISPLY  #LIN6           ;PRINT 'READ CORRECTLY AT OFFSET N'
017442  006301                LIN6.1: ASL     R1              ;DOUBLE THE OFFSET TABLE INDEX
017444  016137  042152  017454        MOV     OFMTBL(R1),1$   ;ADDRESS OF OFFSET POSITION MESSAGE
017452  104422                        DISPLY
017454  000000                1$:     .WORD   0               ;OFFSET VALUE
017456  104422  001157                DISPLY  #$CRLF
017462  000207                        RTS     PC

                                        ;PRINT LINE 6B OF THE ERROR MESSAGE

017464  104422  050141        LINE6B: DISPLY  #LIN6B          ;PRINT 'SECTOR IS ECC CORRECTABLE '
017470  000137  017442                JMP     LIN6.1

                                        ;PRINT LINE 6C OF THE ERROR MESSAGE

017474  104422  050223        LINE6C: DISPLY  #LIN6C          ;'CORRECTED ON NTH RETRY'
017500  005046                LIN6.2: CLR     -(SP)           ;CLEAR STACK
017502  116016  000022                MOVB    $RETRY(R0),(SP) ;RETRY COUNT
017506  004737  016626                JSR     PC,LIN3.2       ;CONVERT & PRINT IT
017512  104422  050241                DISPLY  #LIN6R          ;'RETRY'
017516  104422  001157                DISPLY  #$CRLF
017522  000207                        RTS     PC

                                        ;PRINT LINE 6D OF THE ERROR MESSAGE

017524  104422  050252        LINE6D: DISPLY  #LIN6D          ;'UNCORRECTABLE AFTER N RETRIES'
017530  000137  017500                JMP     LIN6.2          ;FINISH

                                        ;PRINT LINE 7 OF THE ERROR MESSAGE

017534  104422  050325        LINE7:  DISPLY  #LIN7O          ;PRINT ORDER COUNT
017540  012746  000042                MOV     #$OPERC,-(SP)   ;TO STACK
017544  060016                        ADD     R0,(SP)         ;ADD THE BASE ADDRESS
017546  004737  027774                JSR     PC,$D82D        ;CONVERT IT
017552  004737  025510                JSR     PC,$SUPRS       ;PRINT IT
017556  104422  050404                DISPLY  #LIN7T          ;TOTAL ERRORS
017562  016046  000062                MOV     $TOTAL(R0),-(SP)        ;TO STACK
017566  004737  016626                JSR     PC,LIN3.2       ;CONVERT & PRINT
017572  104422  050416                DISPLY  #LIN7X          ;PRINT 'WRDS XFR'
017576  012746  000052                MOV     #$TRANS,-(SP)   ;ADDRESS OF LOW WORD ON STACK
017602  060016                        ADD     R0,(SP)
017604  004737  027774                JSR     PC,$D82D        ;CONVERT
017610  004737  025510                JSR     PC,$SUPRS       ;PRINT
017614  104422  050432                DISPLY  #LIN7R          ;'BITS READ'
017620  012746  000056                MOV     #$READ,-(SP)    ;LOW WORD ADDRESS
017624  060016                        ADD     R0,(SP)
017626  004737  027774                JSR     PC,$D82D        ;CONVERT
017632  004737  025510                JSR     PC,$SUPRS       ;PRINT
017636  104422  001157                DISPLY  #$CRLF
017642  104422  001157                DISPLY  #$CRLF
017646  032737  100000  177570        BIT     #SW15,SWR       ;SEE IF 'HALT ON ERROR' - SWITCH 15
```

```
6188  017654  001401                          BEQ     .+4             ;BR IF NOT
6189  017656  000000                          HALT                    ;SWITCH 15 HALT
6190  017660  000207                          RTS     PC
6191
6192                                  ;PRINT LINE 7A OF ERROR MESSAGE
6193
6194  017662  104422  050325           LINE7A: DISPLY  ,LIN7O          ;'ORDERS = '
6195  017666  012746  000042                   MOV     #$OPERC,-(SP)   ;ORDER COUNT INCREMENT
6196  017672  060016                          ADD     R0,(SP)         ;ADD BASE ADDRESS
6197  017674  004737  027774                   JSR     PC,$D82D        ;CONVERT THE COUNT
6198  017700  004737  025510                   JSR     PC,$SUPRS       ;PRINT IT
6199  017704  104422  050335                   DISPLY  ,LIN7P          ;'TOTAL SEEKS = '
6200  017710  012746  000046                   MOV     #$POSIT,-(SP)   ;TOTAL SEEKS
6201  017714  060016                          ADD     R0,(SP)         ;DEVICE TABLE ADDRESS
6202  017716  004737  027774                   JSR     PC,$D82D        ;CONVERT THE SEEK COUNT
6203  017722  004737  025510                   JSR     PC,$SUPRS       ;PRINT IT
6204  017726  104422  050277                   DISPLY  ,LIN7M          ;'  TOTAL MISPOS ERR = '
6205  017732  016046  000072                   MOV     $MISPO(R0),-(SP)  ;TOTAL ERRORS
6206  017736  004737  016626                   JSR     PC,LIN3.2       ;CONVERT & PRINT IT
6207  017742  104422  050355                   DISPLY  ,LIN7S          ;'  TOTAL SKI,OCYL ERR = '
6208  017746  016046  000070                   MOV     $SKI(R0),-(SP)  ;CONVERT & PRINT IT
6209  017752  004737  016626                   JSR     PC,LIN3.2       ;CONVERT AND PRINT IT
6210  017756  104422  001157                   DISPLY  .$CRLF
6211  017762  104422  001157                   DISPLY  ,$CRLF
6212  017766  032737  100000  177570           BIT     #SW15,SWR       ;SEE IF HALT ON ERROR - SWITCH 15 SET
6213  017774  001401                          BEQ     .+4             ;BR IF NOT
6214  017776  000000                          HALT                    ;SWITCH 13 HALT
6215  020000  000207                          RTS     PC
6216
6217                                  ;PRINT LINE 8 OF THE ERROR MESSAGE
6218
6219  020002  104422  050447           LINE8:  DISPLY  ,LIN8M
6220  020006  004737  015760                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
6221  020012  000207                          RTS     PC
6222
6223                                  ;*****************************************************************
6224
6225                                  .SBTTL  GENERAL SUPPORT SUBROUTINES
6226
6227                                  ;*****************************************************************
6228
6229                                  ;ROUTINE TO INDICATE MEMORY SIZE ERROR AT STARTUP
6230
6231  020014  104400  001157           MEMERR: TYPE    ,$CRLF          ;CR-LF
6232  020020  104400  052140                   TYPE    ,NOTNUF         ;TYPE 'NOT ENOUGH MEMORY'
6233  020024  104400  001157                   TYPE    ,$CRLF          ;CR-LF
6234  020030  000000                          HALT
6235  020032  000137  001430                   JMP     START           ;TRY IT AGAIN
6236
6237                                  ;DECREMENT THE SECTOR-TRACK ADDRESS
6238
6239  020036  005066  000004           READDR: CLR     4(SP)           ;CLEAR STACK FOR SECTOR
6240  020042  116066  000172  000004         MOVB    $RHDA(R0),4(SP) ;INCREMENTED SECTOR ON STACK
6241  020050  105366  000004                 DECB    4(SP)           ;DECREMENT THE SECTOR ADDRESS
```

```
6242  020054  100017                          BPL     1$                      ;BR IF SECTOR GREATER THAN 0
6243  020056  012766  000025  000004          MOV     #25,4(SP)               ;JAM SECTOR ADDRESS TO 21(10)
6244  020064  005066  000002                  CLR     2(SP)                   ;CLEAR STACK FOR TRACK
6245  020070  116066  000173  000002          MOVB    $RHDA+1(R0),2(SP)       ;TRACK ADDRESS
6246  020076  105366  000002                  DECB    2(SP)                   ;DECREMENT TRACK ADDRESS
6247  020102  100007                          BPL     2$                      ;BR IF IT DIDN'T GO NEG
6248  020104  012766  000022  000002          MOV     #22,2(SP)               ;RESET TRACK TO 18(10)
6249  020112  000403                          BR      2$
6250  020114  116066  000173  000002  1$:     MOVB    $RHDA+1(R0),2(SP)       ;TRACK ADDRESS
6251  020122  000207                  2$:     RTS     PC                      ;RETURN
6252
6253                                          ;ROUTINE TO CHECK FOR A PRINTER
6254
6255  020124  012737  177777  001204  CKPTR:  MOV     #-1,$PRFLG              ;CLEAR PRINTER AVAILABILITY FLAG
6256  020132  012737  020156  000004          MOV     #CKPTR1,@#ERRVEC        ;RETURN ADDR IF NO PRINTER
6257  020140  005037  000006                  CLR     @#ERRVEC+2              ;NEW PSW
6258  020144  005777  161036                  TST     @$LSCS                  ;READ PRINTER STATUS REG
6259  020150  005037  001204                  CLR     $PRFLG                  ;SET THE PRINTER AVAILABILITY FLAG
6260  020154  000402                          BR      CKPTR2                  ;FINISHED
6261  020156  062706  000004  CKPTR1:  ADD     #4,SP                   ;RESTORE THE STACK POINTER
6262  020162  012737  000006  000004  CKPTR2:  MOV     #6,@#ERRVEC             ;RESET ERROR VECTOR TO HALT
6263  020170  000207                          RTS     PC
6264
6265                                          ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
6266
6267  020172  012737  177777  001202  CKCLK:  MOV     #-1,CLKFLG              ;CLEAR CLOCK AVAILABILITY FLAG
6268  020200  012737  177777  001200          MOV     #-1,PCLOCK              ;CLEAR KW11-P CLOCK AVAILABILITY FLAG
6269  020206  012737  020266  000004          MOV     #CKCLK1,@#ERRVEC        ;SET UP VECTOR FOR CLOCK CHECK
6270  020214  005037  000006                  CLR     @#ERRVEC+2              ;NEW PSW
6271  020220  005777  160742                  TST     @$LKCSR                 ;CHECK FOR KW11-P
6272  020224  005037  001202                  CLR     CLKFLG                  ;SET CLOCK AVAILABILITY FLAG
6273  020230  005037  001200                  CLR     PCLOCK                  ;SET KW11-P CLOCK FLAG
6274  020234  013701  001172                  MOV     $LPVEC,R1               ;KW11-P VECTOR ADDRESS
6275  020240  012721  021344                  MOV     #CLOCK,(R1)+            ;SET UP KW11-P VECTOR
6276  020244  012711  000300                  MOV     #300,(R1)               ;PSW - PRI 6
6277  020250  012777  177777  160712          MOV     #-1,@$LKCSB             ;LOAD COUNTER BUFFER WITH 1'S
6278  020256  012777  000135  160702          MOV     #135,@$LKCSR            ;SET CLOCK - CNT UP, 16MS, CONT INT
6279  020264  000432                          BR      CKCLK3
6280  020266  062706  000004          CKCLK1:  ADD     #4,SP                   ;RESTORE THE STACK POINTER
6281  020272  012737  020334  000004          MOV     #CKCLK2,@#ERRVEC        ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
6282  020300  005777  160670                  TST     @$LKS                   ;LOOK FOR KW11-L
6283  020304  005037  001202                  CLR     CLKFLG                  ;SET CLOCK FLAG
6284  020310  013701  001176                  MOV     $LLVEC,R1               ;KW11-L VECTOR ADDRESS
6285  020314  012721  021344                  MOV     #CLOCK,(R1)+            ;SET UP KW11-L VECTOR
6286  020320  012711  000300                  MOV     #300,(R1)               ;PSW - PRI 6
6287  020324  012777  000100  160642          MOV     #100,@$LKS              ;SET KW11-L INTERRUPT
6288  020332  000407                          BR      CKCLK3
6289  020334  062706  000004          CKCLK2:  ADD     #4,SP                   ;RESTORE THE STACK POINTER
6290  020340  104400  052165                  TYPE    ,NEDCLK                 ;'P OR L CLOCK MUST BE ON SYSTEM'
6291  020344  000000                          HALT                            ;HALT
6292  020346  000137  001430                  JMP     START                   ;TRY AGAIN
6293  020352  012737  000006  000004  CKCLK3:  MOV     #6,@#ERRVEC             ;RESTORE THE ERROR VECTOR
6294  020360  000207                          RTS     PC
6295
```

```
6296                                    ;ROUTINE TO DISPLAY STATISTICS FOR ALL DRIVES ASSIGNED
6297
6298  020362                            STATPR:
 (2)  020362  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 (2)  020364  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 (2)  020366  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 (2)  020370  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
6299  020372  005004                            CLR     R4              ;R4 CONTAINS THE UNIT POINTER
6300  020374  012703  000010                    MOV     #8.,R3          ;R3 CONTAINS THE UNIT COUNTER
6301  020400  012700  037056                    MOV     #BLK0,R0        ;ADDR OF FIRST COUNTER
6302  020404  105764  041550          1$:       TSTB    ASNLST(R4)      ;SEE IF DRIVE ASSIGNED
6303  020410  001006                            BNE     2$              ;BR IF IT IS
6304  020412  005204                            INC     R4              ;INCREMENT THE UNIT POINTER
6305  020414  062700  000236                    ADD     #$RHEC2+4,R0    ;INCREMENT THE BLOCK ADDRESS
6306  020420  005303                            DEC     R3              ;FINISHED ?
6307  020422  001370                            BNE     1$              ;BR IF NOT
6308  020424  000422                            BR      6$              ;YES, EXIT
6309  020426  004737  020504          2$:       JSR     PC,SHDTYP       ;TYPE THE HEADING
6310  020432  105764  041550          3$:       TSTB    ASNLST(R4)      ;IS THE UNIT ASSIGNED
6311  020436  001003                            BNE     4$              ;BR IF NOT
6312  020440  062700  000236                    ADD     #$RHEC2+4,R0    ;ADD TABLE SIZE TO ADDRESS
6313  020444  000407                            BR      5$
6314  020446  010002                4$:         MOV     R0,R2           ;PUT BLOCK ADDRESS IN R2
6315  020450  062702  000042                    ADD     #$OPERC,R2      ;FIRST COUNTER TO DISPLAY
6316  020454  004737  020556                    JSR     PC,SDETAL       ;TYPE THE STATISTICS
6317  020460  062700  000236                    ADD     #$RHEC2+4,R0    ;INCREMENT BLOCK ADDRESS
6318  020464  005204                5$:         INC     R4              ;INCREMENT UNIT NUMBER
6319  020466  005303                            DEC     R3              ;DECREMENT UNIT COUNTER
6320  020470  001360                            BNE     3$              ;BR IF NOT FINISHED
6321  020472                          6$:
 (2)  020472  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
 (2)  020474  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
 (2)  020476  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
 (2)  020500  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
6322  020502  000207                            RTS     PC
6323
6324                                    ;TYPE THE HEADER FOR THE DRIVE PERFORMANCE SUMMARY TYPEOUT
6325
6326  020504  004737  021314          SHDTYP:   JSR     PC,$TIME        ;TYPE THE TIME OF DAY
6327  020510  004537  025046                    JSR     R5,TYPRI4       ;TYPE AT PRIORITY 4
6328  020514  001157                            $CRLF                   ;CR-LF
6329  020516  004537  025046                    JSR     R5,TYPRI4       ;TYPE THE HEADER
6330  020522  051457                            STATHD                  ;HEADER
6331  020524  005737  001216                    TST     LA30            ;LA30 ON THE SYSTEM ?
6332  020530  001406                            BEQ     1$              ;BR IF ONE IS
6333  020532  004537  025046                    JSR     R5,TYPRI4       ;SPLIT THE HEADER LINE
6334  020536  001157                            $CRLF                   ;CR-LF
6335  020540  004537  025046                    JSR     R5,TYPRI4       ;TYPE SPACES
6336  020544  051272                            LIN4SP                  ;4 SPACES
6337  020546  004537  025046          1$:       JSR     R5,TYPRI4       ;FINISH THE HEADER LINE
6338  020552  051572                            STAHD1                  ;REST OF THE HEADER LINE
6339  020554  000207                            RTS     PC
6340
6341                                    ;TYPE THE ERROR RATE DATA LINE
```

```
 6342
 6343  020556  142737  000007  001255  SDETAL: BICB    #7,UNIT+1       ;CLEAR THE UNIT NUMBER
 6344  020564  150437  001255                  BISB    R4,UNIT+1       ;SET THE UNIT NUMBER
 6345  020570  104400  001254                  TYPE    ,UNIT           ;TYPE IT
 6346  020574  104400  051274                  TYPE    ,LINSP          ;SPACES
 6347  020600  016046  000074                  MOV     $PASSC(R0),-(SP)  ;PUT THE PASS COUNT ON THE STACK
 6348  020604  004737  030170                  JSR     PC,$S82D        ;CONVERT IT
 6349  020610  004737  024550                  JSR     PC,$RPZRS       ;TYPE IT
 6350  020614  104400  051274                  TYPE    ,LINSP          ;SPACES
 6357  020620  010246                          MOV     R2,-(SP)        ;PUT $OPERC ON THE STACK
  (1)  020622  004737  027774                  JSR     PC,$DB2D        ;CONVERT IT
  (1)  020626  004737  024530                  JSR     PC,$RPZR8       ;TYPE $OPERC
  (1)  020632  104400  051274                  TYPE    ,LINSP          ;SPACES
  (1)  020636  062702  000004                  ADD     #4,R2           ;INCREMENT R2
  (1)  020642  010246                          MOV     R2,-(SP)        ;PUT $POSIT ON THE STACK
  (1)  020644  004737  027774                  JSR     PC,$DB2D        ;CONVERT IT
  (1)  020650  004737  024530                  JSR     PC,$RPZR8       ;TYPE $POSIT
  (1)  020654  104400  051274                  TYPE    ,LINSP          ;SPACES
  (1)  020660  062702  000004                  ADD     #4,R2           ;INCREMENT R2
 6364  020664  010246                          MOV     R2,-(SP)        ;PUT $TRANS ON THE STACK
  (1)  020666  004737  027774                  JSR     PC,$DB2D        ;CONVERT $TRANS
  (1)  020672  004737  024560                  JSR     PC,$RPZR0       ;TYPE IT
  (1)  020676  104400  051274                  TYPE    ,LINSP          ;SPACES
  (1)  020702  062702  000004                  ADD     #4,R2           ;INCREMENT R2
 6371  020706  010246                          MOV     R2,-(SP)        ;PUT $READ ON THE STACK
  (1)  020710  004737  027774                  JSR     PC,$DB2D        ;CONVERT $READ
  (1)  020714  004737  024560                  JSR     PC,$RPZR0       ;TYPE IT
   1)  020720  104400  051275                  TYPE    ,LINSP0         ;1 SPACE
  (1)  020724  062702  000004                  ADD     #4,R2           ;INCREMENT R2
 6372  020730  005737  001216                  TST     LA30            ;LA30 ON THE SYSTEM ?
 6373  020734  001406                          BEQ     1$              ;BR IF ONE IS
 6374  020736  104400  001157                  TYPE    ,$CRLF          ;SPLIT THE LINE
 6375  020742  104400  051272                  TYPE    ,LIN4SP         ;4 SPACES
 6376  020746  104400  051274                  TYPE    ,LINSP          ;2 SPACES
 6377  020752  062702  000002          1$:     ADD     #2,R2           ;INCREMENT R2 AGAIN
 6383  020756  012246                          MOV     (R2)+,-(SP)     ;PUT $SOFT ON THE STACK
  (1)  020760  004737  030170                  JSR     PC,$S82D        ;CONVERT $SOFT
  (1)  020764  004737  024540                  JSR     PC,$RPZR4       ;TYPEOUT $SOFT
  (1)  020770  104400  051275                  TYPE    ,LINSP0         ;SPACES
  (1)  020774  012246                          MOV     (R2)+,-(SP)     ;PUT $HARD ON THE STACK
  (1)  020776  004737  030170                  JSR     PC,$S82D        ;CONVERT $HARD
  (1)  021002  004737  024540                  JSR     PC,$RPZR4       ;TYPEOUT $HARD
  (1)  021006  104400  051275                  TYPE    ,LINSP0         ;SPACES
  (1)  021012  012246                          MOV     (R2)+,-(SP)     ;PUT $SKI ON THE STACK
  (1)  021014  004737  030170                  JSR     PC,$S82D        ;CONVERT $SKI
  (1)  021020  004737  024540                  JSR     PC,$RPZR4       ;TYPEOUT $SKI
  (1)  021024  104400  051275                  TYPE    ,LINSP0         ;SPACES
  (1)  021030  012246                          MOV     (R2)+,-(SP)     ;PUT $MISPO ON THE STACK
  (1)  021032  004737  030170                  JSR     PC,$S82D        ;CONVERT $MISPO
  (1)  021036  004737  024540                  JSR     PC,$RPZR4       ;TYPEOUT $MISPO
  (1)  021042  104400  051275                  TYPE    ,LINSP0         ;SPACES
 6384  021046  016046  000062                  MOV     $TOTAL(R0),-(SP)  ;CALCULATE NUMBER OF OTHER ERRORS
 6387  021052  166016  000064                  SUB     $SOFT(R0),(SP)  ;SUBTRACT $SOFT FROM $TOTAL
  (1)  021056  166016  000066                  SUB     $HARD(R0),(SP)  ;SUBTRACT $HARD FROM $TOTAL
```

```
   (1)  021062  166016  000070                SUB     $SKI(R0),(SP)    ;SUBTRACT $SKI FROM $TOTAL
   (1)  021066  166016  000072                SUB     $MISPO(R0),(SP)  ;SUBTRACT $MISPO FROM $TOTAL
 6388   021072  004737  030170                JSR     PC,$S82D         ;CONVERT 'OTHER' COUNT
 6389   021076  004737  024540                JSR     PC,$RPZR4        ;TYPE IT
 6390   021102  104400  001157                TYPE    ,$CRLF
 6391   021106  000207                        RTS     PC
 6392
 6393                                ;ROUTINE TO TYPE STATISTICS FOR THE DRIVE IN R0
 6394
 6395   021110                       TYPEST:
   (2)  021110  010046                        MOV     R0,-(SP)         ;;PUSH R0 ON STACK
   (2)  021112  010246                        MOV     R2,-(SP)         ;;PUSH R2 ON STACK
   (2)  021114  010446                        MOV     R4,-(SP)         ;;PUSH R4 ON STACK
 6396   021116  004737  020504                JSR     PC,SHDTYP        ;TYPE THE HEADING
 6397   021122  005004                        CLR     R4               ;CLEAR R4 FOR DRIVE NUMBER
 6398   021124  111004                        MOVB    (R0),R4          ;DRIVE NUMBER
 6399   021126  010002                        MOV     R0,R2            ;PUT BLOCK ADDRESS IN R2
 6400   021130  062702  000042                ADD     #$OPERC,R2       ;ADDRESS OF '$OPERC' IN PRESENT CLOCK
 6401   021134  004737  020556                JSR     PC,SDETAL        ;TYPE THE STATISTICS
 6402   021140  012604                        MOV     (SP)+,R4         ;;POP STACK INTO R4
   (2)  021142  012602                        MOV     (SP)+,R2         ;;POP STACK INTO R2
   (2)  021144  012600                        MOV     (SP)+,R0         ;;POP STACK INTO R0
 6403   021146  000207                        RTS     PC               ;RETURN
 6418
 6419
   (1)                                ;ROUTINE TO INCREMENT $SOFT
   (1)                                ;
   (1)                                ;NOTE:  $SOFT WILL NOT BE INCREMENTED BEYOND 9999 (10)
   (1)
   (1)  021150  005737  016330       INCSOF: TST     PRT2B            ;SEE IF BAD TRK/SEC INDICATOR SET
   (1)  021154  001006                        BNE     1$               ;BR IF IT'S SET, DON'T INCREMENT COUNT
   (1)  021156  026027  000064  023417        CMP     $SOFT(R0),#9999. ;IS $SOFT ALREADY AT MAXIMUM ?
   (1)  021164  103002                        BHIS    1$               ;BR IF IT IS
   (1)  021166  005260  000064                INC     $SOFT(R0)                ;INCREMENT $SOFT
   (1)  021172  000207                 1$:    RTS     PC               ;RETURN
   (1)
 6420
   (1)                                ;ROUTINE TO INCREMENT $HARD
   (1)                                ;
   (1)                                ;NOTE:  $HARD WILL NOT BE INCREMENTED BEYOND 9999 (10)
   (1)
   (1)  021174  005737  016330       INCHRD: TST     PRT2B            ;SEE IF BAD TRK/SEC INDICATOR SET
   (1)  021200  001006                        BNE     1$               ;BR IF IT'S SET, DON'T INCREMENT COUNT
   (1)  021202  026027  000066  023417        CMP     $HARD(R0),#9999. ;IS $HARD ALREADY AT MAXIMUM ?
   (1)  021210  103002                        BHIS    1$               ;BR IF IT IS
   (1)  021212  005260  000066                INC     $HARD(R0)                ;INCREMENT $HARD
   (1)  021216  000207                 1$:    RTS     PC               ;RETURN
   (1)
 6421
   (1)                                ;ROUTINE TO INCREMENT $SKI
   (1)                                ;
   (1)                                ;NOTE:  $SKI WILL NOT BE INCREMENTED BEYOND 9999 (10)
   (1)
   (1)  021220  005737  016330       INCSKI: TST     PRT2B            ;SEE IF BAD TRK/SEC INDICATOR SET
```

```
   (1)  021224  001006                        BNE     1$                ;BR IF IT'S SET, DON'T INCREMENT COUNT
   (1)  021226  026027  000070  023417        CMP     $SKI(R0),#9999.   ;IS $SKI ALREADY AT MAXIMUM ?
   (1)  021234  103002                        BHIS    1$                ;BR IF IT IS
   (1)  021236  005260  000070                INC     $SKI(R0)                  ;INCREMENT $SKI
   (1)  021242  000207                1$:     RTS     PC                ;RETURN
   (1)
  6422
   (1)                                        ;ROUTINE TO INCREMENT $MISPO
   (1)                                        ;
   (1)                                        ;NOTE:  $MISPO WILL NOT BE INCREMENTED BEYOND 9999 (10)
   (1)
   (1)  021244  005737  016330        INCMIS. TST     PRT2B             ;SEE IF BAD TRK/SEC INDICATOR SET
   (1)  021250  001006                        BNE     1$                ;BR IF IT'S SET, DON'T INCREMENT COUNT
   (1)  021252  026027  000072  023417        CMP     $MISPO(R0),#9999. ;IS $MISPO ALREADY AT MAXIMUM ?
   (1)  021260  103002                        BHIS    1$                ;BR IF IT IS
   (1)  021262  005260  000072                INC     $MISPO(R0)                ;INCREMENT $MISPO
   (1)  021266  000207                1$:     RTS     PC                ;RETURN
   (1)
  6423
   (1)                                        ;ROUTINE TO INCREMENT $TOTAL
   (1)                                        ;
   (1)                                        ;NOTE:  $TOTAL WILL NOT BE INCREMENTED BEYOND 9999 (10)
   (1)
   (1)  021270  005737  016330        INCTOT: TST     PRT2B             ;SEE IF BAD TRK/SEC INDICATOR SET
   (1)  021274  001006                        BNE     1$                ;BR IF IT'S SET, DON'T INCREMENT COUNT
   (1)  021276  026027  000062  023417        CMP     $TOTAL(R0),#9999. ;IS $TOTAL ALREADY AT MAXIMUM ?
   (1)  021304  103002                        BHIS    1$                ;BR IF IT IS
   (1)  021306  005260  000062                INC     $TOTAL(R0)                ;INCREMENT $TOTAL
   (1)  021312  000207                1$:     RTS     PC                ;RETURN
   (1)
  6424
  6425                                        ;ROUTINE TO TYPE THE TIME
  6426
  6427  021314  005737  001202        $TIME:  TST     CLKFLG            ;CLOCK ON THE SYSTEM ?
  6428  021320  001010                        BNE     1$                ;BR IF NOT
  6429  021322  104400  001157                TYPE    ,$CRLF
  6430  021326  104400  021614                TYPE    ,HOUR             ;TYPE THE HOURS
  6431  021332  104400  021620                TYPE    ,MINUTE           ;TYPE THE MINUTES
  6432  021336  104400  021624                TYPE    ,SECOND           ;TYPE THE SECONDS
  6433  021342  000207                1$:     RTS     PC
  6434
  6435                                        ;CLOCK HANDLER ROUTINE
  6436
  6437  021344  010046                CLOCK:  MOV     R0,-(SP)          ;SAVE R0
  6438  021346  005237  021632                INC     SIXTEE            ;INCREMENT THE 1/60 SECOND COUNTER
  6439  021352  001076                        BNE     1$                ;BR IF A SECOND NOT COUNTED
  6440  021354  013737  001214  021632        MOV     HZ,SIXTEE         ;RESTORE THE VALUE
  6441  021362  005437  021632                NEG     SIXTEE            ;MAKE IT INTO 2'S COMP
  6442  021366  105237  021625                INCB    SECOND+1          ;INCREMENT THE SECONDS
  6443  021372  122737  000072  021625        CMPB    #72,SECOND+1      ;SEE IF COUNTER AT MAX
  6444  021400  001063                        BNE     1$                ;BR IF NOT
  6445  021402  112737  000060  021625        MOVB    #60,SECOND+1      ;RESTORE VALUE
  6446  021410  105237  021624                INCB    SECOND            ;INCREMENT UPPER
  6447  021414  122737  000066  021624        CMPB    #66,SECOND        ;UPPER AT MAX
```

```
 6448  021422  001052                          BNE     1$              ;BR IF NOT
 6449  021424  112737  000060  021624          MOVB    #60,SECOND      ;RESET UPPER
 6450  021432  105337  001277                  DECB    INTRVL+1        ;DECREMENT STATISTICS TYPEOUT INTERVAL
 6451  021436  105237  021621                  INCB    MINUTE+1        ;INCREMENT MINUTE
 6452  021442  122737  000072  021621          CMPB    #72,MINUTE+1    ;AT MAX ?
 6453  021450  001037                          BNE     1$              ;BR IF NOT
 6454  021452  112737  000060  021621          MOVB    #60,MINUTE+1    ;RESET LOWER
 6455  021460  105237  021620                  INCB    MINUTE          ;INCREMENT UPPER HALF
 6456  021464  122737  000066  021620          CMPB    #66,MINUTE      ;AT MAX ?
 6457  021472  001026                          BNE     1$              ;BR IF NOT
 6458  021474  112737  000060  021620          MOVB    #60,MINUTE      ;RESET UPPER
 6459  021502  105237  021615                  INCB    HOUR+1          ;INCREMENT HOURS
 6460  021506  022737  035071  021614          CMP     #35071,HOUR     ;AT MAX (99) ?
 6461  021514  001412                          BEQ     2$              ;BR IF IT IS
 6462  021516  122737  000072  021615          CMPB    #72,HOUR+1      ;AT MAX ?
 6463  021524  001011                          BNE     1$              ;BR IF NOT
 6464  021526  112737  000060  021615          MOVB    #60,HOUR+1      ;RESET VALUE
 6465  021534  105237  021614                  INCB    HOUR            ;INCREMENT UPPER
 6466  021540  000403                          BR      1$
 6467  021542  012737  030060  021614  2$:     MOV     #30060,HOUR     ;RESET HOUR FIELD
 6468  021550  012746  000021          1$:     MOV     #21,-(SP)       ;17 MS ON THE STACK
 6469  021554  004037  035036                  JSR     R0,RPTMR        ;DRIVER TIMER ROUTINE
 6470  021560  105737  001276                  TSTB    INTRVL          ;DISPLAY THE PERFORMANCE SUMMARY ?
 6471  021564  001411                          BEQ     3$              ;BR IF NOT
 6472  021566  105737  001277                  TSTB    INTRVL+1        ;DISPLAY INTERVAL FINISHED ?
 6473  021572  001006                          BNE     3$              ;BR IF NOT
 6474  021574  113737  001276  001277          MOVB    INTRVL,INTRVL+1 ;RESTORE THE INTERVAL
 6475  021602  012737  177777  001222          MOV     #-1,STATIN      ;SET ERROR RATE DATA DISPLAY FLAG
 6476  021610  012600                  3$:     MOV     (SP)+,R0        ;RESTORE R0
 6477  021612  000002                          RTI
 6478
 6479  021614     060     060          HOUR:   .BYTE   60,60           ;HOURS
 6480  021616     072     000                  .BYTE   72,0            ;':' & TERMINATOR
 6481  021620     060     060          MINUTE: .BYTE   60,60           ;MINUTES
 6482  021622     072     000                  .BYTE   72,0            ;':' & TERMINATOR
 6483  021624     060     060          SECOND: .BYTE   60,60           ;SECONDS
 6484  021626     040     040                  .BYTE   40,40           ;SPACES
 6485  021630     000                          .BYTE   0               ;TERMINATOR
 6486          021632                          .EVEN
 6487
 6488  021632  000000                  SIXTEE: .WORD   0               ;1/60TH OR 1/50TH OF A SECOND COUNTER
 6489
 6490                                  ;COMMAND DECODE ROUTINE
 6491
 6492  021634                          KSR:
  (2)  021634  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  021636  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  021640  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  021642  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  021644  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  021646  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 6493  021650  012705  026177                  MOV     #$TKQSRT+1,R5   ;ADDR OF INPUT
 6494  021654  104400  001157                  TYPE    ,$CRLF          ;CR-LF
 6495  021660  122715  000101                  CMPB    #101,(R5)       ;EQ TO AN 'A'
```

```
 6496  021664  001410                          BEQ     1$              ;BR IF IT IS
 6497  021666  121527  000075                  CMPB    (R5),#75        ;UNIT NUMBER GREATER THAN AN ASCII 7 ?
 6498  021672  103054                          BHIS    6$              ;BR IF IT IS
 6499  021674  121527  000060                  CMPB    (R5),#60        ;UNIT NUMBER LESS THAN AN ASCII 0 ?
 6500  021700  103451                          BLO     6$              ;BR IF IT IS
 6501  021702  142715  177770                  BICB    #^C7,(R5)       ;LEAVE ONLY LOWER 3 BITS IF CHAR NOT 'A'
 6502  021706  122765  000124  177777  1$:     CMPB    #'T,-1(R5)      ;EQ TO 'T'
 6503  021714  001003                          BNE     2$              ;BR IF NOT EQ
 6504  021716  004737  022046                  JSR     PC,NEWASN       ;ASSIGN UNIT FOR TEST
 6505  021722  000442                          BR      7$              ;EXIT
 6506  021724  122765  000104  177777  2$:     CMPB    #'D,-1(R5)      ;EQ TO 'D' ?
 6507  021732  001003                          BNE     3$              ;BR IF NOT EQ
 6508  021734  004737  022364                  JSR     PC,DEASGN       ;DEASSIGN UNIT
 6509  021740  000433                          BR      7$              ;EXIT
 6510  021742  122765  000123  177777  3$:     CMPB    #'S,-1(R5)      ;EQ TO 'S'
 6511  021750  001003                          BNE     4$              ;BR IF NOT EQ
 6512  021752  004737  022466                  JSR     PC,SCMND        ;TYPE STATISTICS
 6513  021756  000424                          BR      7$              ;EXIT
 6514  021760  122765  000127  177777  4$:     CMPB    #'W,-1(R5)      ;EQ TO 'W'
 6515  021766  001007                          BNE     5$              ;BR IF NOT EQ
 6516  021770  032737  000001  177570          BIT     #SW0,SWR        ;IS SWITCH 0 SET ?
 6517  021776  001012                          BNE     6$              ;BR IF SET, CAN'T DO 'W' COMMAND
 6518  022000  004737  022740                  JSR     PC,DATAPK       ;WRITE A DATA PACK
 6519  022004  000411                          BR      7$              ;EXIT
 6520  022006  122765  000122  177777  5$:     CMPB    #'R,-1(R5)      ;EQ TO 'R' ?
 6521  022014  001003                          BNE     6$              ;BR IF NOT EQ
 6522  022016  004737  022752                  JSR     PC,REDAPK       ;READ A DATA PACK
 6523  022022  000402                          BR      7$              ;EXIT
 6524  022024  104400  052315          6$:     TYPE    ,INVLD          ;TYPE 'INVALID COMMAND' MESSAGE
 6525  022030                            7$:
  (2)  022030  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
  (2)  022032  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2)  022034  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)  022036  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  022040  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  022042  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 6526  022044  000207                          RTS     PC
 6527
 6528                          ;ROUTINE TO ASSIGN A DRIVE ('T' COMMAND)
 6529
 6530  022046  005037  001224          NEWASN: CLR     PACK            ;CLEAR 'W' COMMAND INDICATOR
 6531  022052  005004                  ASGN0:  CLR     R4              ;R4 IS TABLE INDEX
 6532  022054  012703  000010                  MOV     #8.,R3          ;SET INITIAL COUNT TO 8
 6533  022060  122715  000101                  CMPB    #101,(R5)       ;ASSIGN ALL UNITS?
 6534  022064  001416                          BEQ     ASGN2           ;BR IF ALL UNITS
 6535  022066  111504                  ASGN1:  MOVB    (R5),R4         ;PUT UNIT # IN R4
 6536  022070  012737  051347  023524          MOV     #UNTASN,ASNMSG  ;'UNIT ASSIGNED' MESSAGE ADDR
 6537  022076  012703  000001                  MOV     #1,R3           ;RELOAD R3 FOR 1 UNIT
 6538  022102  105764  041550                  TSTB    ASNLST(R4)      ;UNIT ALREADY ASSIGNED
 6539  022106  001002                          BNE     1$              ;BR IF IT IS
 6540  022110  004737  022144                  JSR     PC,ASGN3        ;SEE IF UNIT ON THE SYSTEM
 6541  022114  000137  023474          1$:     JMP     ASNERR          ;RETURN HERE IF DRIVE NOT AVAIL
 6542  022120  000207                          RTS     PC              ;EXIT
 6543  022122  004737  022144          ASGN2:  JSR     PC,ASGN3        ;ASSIGN ALL UNASSIGNED, AVAIL UNITS
```

```
6544  022126  000137  022134              JMP     1$              ;UNIT NOT ON SYSTEM
6545  022132  000207                      RTS     PC              ;EXIT
6546  022134  012746  022126      1$:     MOV     #ASGN2+4,-(SP)  ;PUT RETURN ADDRESS ON THE STACK
6547  022140  000137  022254              JMP     ASGN4           ;LOOK FOR MORE UNITS
6548  022144  105764  030550      ASGN3:  TSTB    ASNLS+1(R4)     ;UNIT ALREADY ASSIGNED?
6549  022150  001041                      BNE     ASGN4           ;BR IF IT IS
6550  022152  005737  030610      1$:     TST     DTJW            ;DATA TRANSFER UNDER WAY ?
6551  022156  100375                      BPL     1$              ;BR IF IT IS
6552  022160  110437  041436              MOVB    R4,GENDPB       ;DRIVE NUMBER
6553  022164  004737  012724              JSR     PC,RECALO       ;RECALIBRATE DRIVE
6554  022170  105764  030466              TSTB    DRVSTA(R4)      ;DRIVE AVAILABLE?
6555  022174  001436                      BEQ     ASGN6           ;BR IF DRIVE OFFLINE
6556  022176  100441                      BMI     ASGN7           ;BR DRIVE NOT AVAILABLE
6557  022200  006304                      ASL     R4              ;MAKE R4 INTO WORD INDEX
6558  022202  016464  042034  041602      MOV     BLKADR(R4),NEWUNIT(R4) ;DPB ADDRESS
6559  022210  016400  042034              MOV     BLKADR(R4),R0   ;PUT BLOCK'S ADDR INTO R0
6560  022214  004737  013204              JSR     PC,CLRDPB       ;CLEAR BLOCK FOR UNIT JUST ASSIGNED
6561  022220  004737  022764              JSR     PC,GETID        ;GET DRIVE I.D.
6562  022224  004737  023044              JSR     PC,GETADR       ;GET BAD SECTOR ADDRESSES
6563  022230  012760  000001  000074      MOV     #1,$PASSC(R0)   ;PRESET PASS COUNT TO 1
6564  022236  005737  001224              TST     PACK            ;WRITE DATA PACK ?
6565  022242  001403                      BEQ     2$              ;BR IF NOT
6566  022244  113760  001224  000031      MOVB    PACK,$PACK(R0)  ;SET READ/WRITE DATA PACK INDICATOR
6567  022252  006204              2$:     ASR     R4              ;RESTORE UNIT ADDRESS
6568  022254  005303              ASGN4:  DEC     R3              ;DEC UNIT COUNT
6569  022256  001405                      BEQ     ASGN5           ;BR IF FINISHED
6570  022260  005204                      INC     R4              ;INCREMENT UNIT NUMBER
6571  022262  000730                      BR      ASGN3           ;CONTINUE
6572  022264  062716  000004      ASGN5:  ADD     #4,(SP)         ;INCREMENT RETURN
6573  022270  000207                      RTS     PC              ;RETURN
6574  022272  012737  051304  023524  ASGN6:  MOV     #UNTOFF,ASNMSG  ;'OFFLINE' MESSAGE ADDRESS
6575  022300  000207                      RTS     PC              ;RETURN
6576  022302  006304              ASGN7:  ASL     R4              ;R4 INTO WORD INDEX
6577  022304  005764  030476              TST     DRVTYP(R4)      ;UNIT PRESENT?
6578  022310  001414                      BEQ     1$              ;BR IF NOT
6579  022312  022764  020020  030476      CMP     #20020,DRVTYP(R4) ;UNIT RP04?
6580  022320  001414                      BEQ     2$              ;BR IF SINGLE PORT RP04
6581  022322  022764  024020  030476      CMP     #24020,DRVTYP(R4) ;UNIT DUAL PORT RP04?
6582  022330  001410                      BEQ     2$              ;BR IF DUAL PORT RP04
6583  022332  012737  051375  023524      MOV     #NOTRP,ASNMSG   ;ADDRESS OF 'NOT RP04' MSG
6584  022340  000407                      BR      3$              ;EXIT
6585  022342  012737  051412  023524  1$:     MOV     #NOTPRS,ASNMSG  ;ADDRESS OF 'NOT PRESENT' MSG
6586  022350  000403                      BR      3$              ;EXIT
6587  022352  012737  051427  023524  2$:     MOV     #NOTSAF,ASNMSG  ;ADDR OF 'UNIT UNSAFE' MESSAGE
6588  022360  006204              3$:     ASR     R4              ;RESTORE R4
6589  022362  000207                      RTS     PC              ;ERROR RETURN
6590
6591                          ;ROUTINE TO DEASSIGN A UNIT ('D' COMMAND)
6592
6593  022364  005004          DEASGN: CLR     R4
6594  022366  122715  000101              CMPB    #101,(R5)       ;DEASSIGN ALL UNITS ?
6595  022372  001432                      BEQ     5$              ;BR IF YES
6596  022374  012703  000001              MOV     #BIT00,R3       ;SET R3 FOR ONE UNIT
6597  022400  111504                      MOVB    (R5),R4         ;GET UNIT NUMBER
```

```
6598  022402  105764  041550          1$:     TSTB    ASNLST(R4)          ;UNIT ASSIGNED ?
6599  022406  001413                          BEQ     3$                  ;BR IF NOT
6600  022410  105064  041550                  CLRB    ASNLST(R4)          ;DELETE THE UNIT FROM THE ASSIGNED LIST
6601  022414  006304                          ASL     R4                  ;MAKE ADDR INTO A WORD INDEX
6602  022416  016464  042034  041560          MOV     BLKADR(R4),DUNIT(R4)  ;PUT ADDRESS IN DEASSIGN LIST
6603  022424  006204                          ASR     R4
6604  022426  005303                  2$:     DEC     R3                  ;ANY MORE UNITS ?
6605  022430  001412                          BEQ     4$                  ;BR IF NOT
6606  022432  005204                          INC     R4
6607  022434  000762                          BR      1$
6608  022436  122715  000101          3$:     CMPB    #101,(R5)           ;DEASSIGN ALL UNITS ?
6609  022442  001771                          BEQ     2$                  ;BR IF YES
6610  022444  012737  051325  023524          MOV     #UNTNOT,ASNMSG      ;ADDR OF 'NOT ASSIGNED' MESSAGE
6611  022452  004737  023474                  JSR     PC,ASNERR           ;REPORT IT
6612  022456  000207                  4$:     RTS     PC
6613  022460  012703  000010          5$:     MOV     #8.,R3              ;SET UNIT COUNT TO 8
6614  022464  000746                          BR      1$
6615
6616                                  ;ROUTINE TO TYPE UNIT PERFORMANCE SUMMARY  ('S' COMMAND)
6617
6618  022466  005004                  SCMND:  CLR     R4
6619  022470  122715  000101                  CMPB    #101,(R5)           ;ALL STATISTICS ?
6620  022474  001420                          BEQ     2$                  ;BR IF YES
6621  022476  111504                          MOVB    (R5),R4             ;GET UNIT #
6622  022500  105764  041550                  TSTB    ASNLST(R4)          ;SEE IF UNIT ASSIGNED
6623  022504  001406                          BEQ     1$                  ;BR IF NOT
6624  022506  006304                          ASL     R4                  ;MAKE UNIT ADDR INTO WORD INDEX
6625  022510  016400  042034                  MOV     BLKADR(R4),R0       ;ADDR OF BLOCK
6626  022514  004737  021110                  JSR     PC,TYPEST           ;TYPE DRIVE STATISTICS
6627  022520  000506                          BR      9$                  ;EXIT
6628  022522  012737  051325  023524  1$:     MOV     #UNTNOT,ASNMSG      ;ADDR OF 'NOT ASSIGNED' MSG
6629  022530  004737  023474                  JSR     PC,ASNERR           ;TYPE ERROR MESSAGE
6630  022534  000500                          BR      9$                  ;EXIT
6631  022536  012703  000010          2$:     MOV     #8.,R3              ;UNIT COUNT
6632  022542  105764  041550          3$:     TSTB    ASNLST(R4)          ;SEE IF ANY UNIT ASSIGNED
6633  022546  001004                          BNE     4$
6634  022550  005204                          INC     R4                  ;INCREMENT UNIT ADDRESS
6635  022552  005303                          DEC     R3                  ;DECREMENT COUNTER
6636  022554  001372                          BNE     3$                  ;MORE TO CHECK
6637  022556  000467                          BR      9$                  ;NONE ASSINGED, RETURN
6638  022560  004737  020362          4$:     JSR     PC,STATPR           ;TYPE ALL STATISTICS
6639  022564  105737  001226                  TSTB    DATE                ;SEE IF 'DATE' ENTERED
6640  022570  001404                          BEQ     11$                 ;BR IF NOT
6641  022572  104400  052436                  TYPE    ,DATEIS             ;'DATE: '
6642  022576  104400  001226                  TYPE    ,DATE               ;THE OPERATOR ENTERED DATE
6643  022602  105737  001240          11$:    TSTB    OPERID              ;SEE IF OPERATOR I.D. ENTERED
6644  022606  001404                          BEQ     12$                 ;BR IF NOT
6645  022610  104400  052447                  TYPE    ,IDIS               ;'OPERATOR I.D.: '
6646  022614  104400  001240                  TYPE    ,OPERID             ;THE OPERATOR I.D.
6647  022620  104400  052471          12$:    TYPE    ,HEDLIN             ;HEADER LINE
6648  022624  012737  037232  022706          MOV     #BLK0+$DRVID,6$     ;DRIVE I.D. FIELD ADDRESS
6649  022632  005004                          CLR     R4                  ;DRIVE ADDRESS
6650  022634  012703  000010                  MOV     #8.,R3              ;COUNTER
6651  022640  105764  041550          5$:     TSTB    ASNLST(R4)          ;SEE IF DRIVE ASSIGNED
```

```
6652  022644  001423                          BEQ     7$              ;BR IF NOT ASSIGNED
6653  022646  142737  000007  001255          BICB    #7,UNIT+1       ;CLEAR OLD DRIVE NUMBER
6654  022654  150437  001255                  BISB    R4,UNIT+1       ;LOAD NEW DRIVE NUMBER
6655  022660  104400  001254                  TYPE    ,UNIT           ;TYPE THE DRIVE NUMBER
6656  022664  104400  051272                  TYPE    ,LIN4SP         ;4 SPACES
6657  022670  105777  000012                  TSTB    @6$             ;SEE IF DRIVE I.D. ENTERED
6658  022674  001003                          BNE     10$             ;BR IF DRIVE I.D. PRESENT
6659  022676  104400  052514                  TYPE    ,NONE           ;TYPE 'NONE'
6660  022702  000404                          BR      7$              ;CONTINUE
6661  022704  104400                  10$:    TYPE                    ;TYPE THE DRIVE I.D.
6662  022706  000000                  6$:     .WORD   0               ;ADDRESS OF DRIVE I.D. FIELD HERE
6663  022710  104400  001157                  TYPE    ,$CRLF          ;CR-LF
6664  022714  005303                  7$:     DEC     R3              ;DECREMENT THE COUNTER
6665  022716  001405                          BEQ     8$              ;BR IF AT END
6666  022720  062737  000236  022706          ADD     #$RHEC2+4,6$    ;INCREMENT THE MESSAGE FIELD ADDRESS
6667  022726  005204                          INC     R4              ;INCREMENT DRIVE ADDRESS
6668  022730  000743                          BR      5$              ;CONTINUE
6669  022732  104400  001157          8$:     TYPE    ,$CRLF          ;CR-LF
6670  022736  000207                  9$:     RTS     PC
6671
6672                                  ;ROUTINE TO WRITE A DATA PACK ('W' COMMAND)
6673
6674  022740  012737  177777  001224  DATAPK: MOV     #-1,PACK        ;SET THE 'W' COMMAND INDICATOR
6675  022746  000137  022052                  JMP     ASGNO           ;ASSIGN REQUESTED UNIT
6676
6677
6678                                  ;ROUTINE TO READ A DATA PACK ('R' COMMAND)
6679
6680  022752  012737  000001  001224  REDAPK: MOV     #1,PACK         ;SET THE 'READ' INDICATOR
6681  022760  000137  022052                  JMP     ASGNO           ;ASSIGN THE REQUESTED UNIT
6682
6683                                  ;ROUTINE TO GET THE DRIVE ID NUMBER FROM THE OPERATOR
6684
6685  022764  010546                  GETID:  MOV     R5,-(SP)        ;SAVE R5
6686  022766  104400  052404                  TYPE    ,ENTDRV         ;'ENTER DRV I.D.: '
6687  022772  142737  000007  001255          BICB    #7,UNIT+1       ;CLEAR OLD UNIT NUMBER
6688  023000  010446                          MOV     R4,-(SP)        ;CONVERT INDEX TO DRIVE NUMBER
6689  023002  006216                          ASR     (SP)            ;SHIFT IT
6690  023004  152637  001255                  BISB    (SP)+,UNIT+1    ;LOAD NEW UNIT NUMBER
6691  023010  104400  001254                  TYPE    ,UNIT           ;TYPE THE DRIVE NUMBER
6692  023014  104400  052433                  TYPE    ,COLON          ;TYPE A COLON ':'
6693  023020  104414                          RDLIN                   ;READ THE ENTRY
6694  023022  012605                          MOV     (SP)+,R5        ;GET THE ENTRY ADDRESS
6695  023024  012560  000154                  MOV     (R5)+,$DRVID(R0)  ;STORE THE I.D.
6696  023030  012560  000156                  MOV     (R5)+,$DRVID+2(R0)  ;STORE THE I.D.
6697  023034  012560  000160                  MOV     (R5)+,$DRVID+4(R0)  ;STORE THE I.D.
6698  023040  012605                          MOV     (SP)+,R5        ;RESTORE R5
6699  023042  000207                          RTS     PC              ;RETURN
6700
6701                                  ;ROUTINE TO GET THE ADDRESSES OF ANY BAD SECTORS (UP TO A MAX OF 8)
6702
6703  023044                          GETADR:
 (2)  023044  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (2)  023046  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
```

```
  (2) 023050 010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 6704 023052 104400 052523                   TYPE    ,ENTADR         ;ENTER SECTOR ADDRESSES
 6705 023056 104400 001254                   TYPE    ,UNIT           ;TYPE THE UNIT NUMBER
 6706 023062 104400 051275                   TYPE    ,LINSPO         ;1 SPACE
 6707 023066 104400 001156                   TYPE    ,$QUES          ;QUESTION MARK
 6708 023072 012703 000020                   MOV     #16.,R3         ;NUMBER OF LOCATIONS IN THE TABLE TO PRESET
 6709 023076 012701 000114                   MOV     #$BDSEC,R1      ;TABLE INCREMENT
 6710 023102 060001                          ADD     R0,R1           ;BLOCK STARTING ADDRESS
 6711 023104 012721 177777          1$:      MOV     #-1,(R1)+       ;SET LOCATION TO 1'S
 6712 023110 005303                          DEC     R3              ;DECREMENT TABLE SIZE COUNT
 6713 023112 001374                          BNE     1$              ;BR IF NOT FINISHED WITH TABLE
 6714 023114 162701 000040                   SUB     #32.,R1         ;SET POINTER TO BEGINNING OF TABLE
 6715 023120 012703 000010                   MOV     #8.,R3          ;NUMBER OF ADDRESSES IN TABLE
 6716 023124 104414                 2$:      RDLIN                   ;GET ADDRESS FROM OPERATOR
 6717 023126 012602                          MOV     (SP)+,R2        ;TEXT POINTER
 6718 023130 121227 000003                   CMPB    (R2),#3         ;'CONTROL C' ENTERED ?
 6719 023134 001476                          BEQ     4$              ;BR IF IT WAS
 6720 023136 012737 000054 023472            MOV     #',,SEPART      ;COMMA IS THE SEPARATOR
 6721 023144 004537 023344                   JSR     R5,CONVEA       ;GET THE CYLINDER ADDRESS
 6722 023150 000460                          BR      3$              ;TERMINATOR FOUND (INVALID)
 6723 023152 005737 023470                   TST     HOLD            ;SEE IF INVALID TEXT ENTERED
 6724 023156 100455                          BMI     3$              ;BR IF ANY CYLINDER TEXT INVALID
 6725 023160 023727 023470 000632            CMP     HOLD,#410.      ;IS CYLINDER VALUE OK ?
 6726 023166 101051                          BHI     3$              ;BR IF NOT
 6727 023170 013711 023470                   MOV     HOLD,(R1)       ;MOVE CYLINDER VALUE TO THE TABLE
 6728 023174 005037 023342                   CLR     6$              ;CLEAR THE TRACK 'CR' INDICATOR
 6729 023200 004537 023344                   JSR     R5,CONVEA       ;GET THE TRACK ADDRESS
 6730 023204 000402                          BR      .+6             ;'CR' ENTERED; ENTIRE TRACK SPECIFIED
 6731 023206 005237 023342                   INC     6$              ;SET THE 'SECTOR' SWITCH
 6732 023212 005737 023470                   TST     HOLD            ;SEE IF INVALID CHARACTERS ENTERED
 6733 023216 100435                          BMI     3$              ;BR IF ANY WERE
 6734 023220 023727 023470 000022            CMP     HOLD,#18.       ;SEE IF TRACK ADDRESS TOO LARGE
 6735 023226 101031                          BHI     3$              ;BR IF IT IS
 6736 023230 113761 023470 000003            MOVB    HOLD,3(R1)      ;PUT TRACK ADDRESS INTO TABLE
 6737 023236 005737 023342                   TST     6$              ;SEE IF <CR> ENTERED
 6738 023242 001416                          BEQ     5$              ;GET OUT IF IT WAS
 6739 023244 004537 023344                   JSR     R5,CONVEA       ;GET SECTOR ADDRESS
 6740 023250 000401                          BR      .+4             ;<CR> MUST BE ENTERED
 6741 023252 000417                          BR      3$              ;INVALID, SEPARATOR USED
 6742 023254 005737 023470                   TST     HOLD            ;SEE IF INVALID CHARACTER ENTERED
 6743 023260 100414                          BMI     3$              ;BR IF ONE WAS
 6744 023262 023727 023470 000025            CMP     HOLD,#21.       ;SEE IF SECTOR ADDRESS TOO LARGE
 6745 023270 101010                          BHI     3$              ;BR IF IT IS
 6746 023272 113761 023470 000002            MOVB    HOLD,2(R1)      ;PUT SECTOR ADDRESS INTO TABLE
 6747 023300 005303                 5$:      DEC     R3              ;MORE ENTRIES ?
 6748 023302 001413                          BEQ     4$              ;BR IF NOT
 6749 023304 062701 000004                   ADD     #4,R1           ;INCREMENT THE TABLE POINTER
 6750 023310 000705                          BR      2$              ;CONTINUE
 6751 023312 012711 177777          3$:      MOV     #-1,(R1)        ;CLEAR PRESENT TABLE ENTRY
 6752 023316 012761 177777 000002            MOV     #-1,2(R1)       ;CLEAR PRESENT TABLE ENTRY
 6753 023324 104400 052556                   TYPE    ,BADENT         ;'INVALID ENTRY'
 6754 023330 000675                          BR      2$              ;TRY AGAIN
 6755 023332                        4$:
  (2) 023332 012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
```

```
   (2)  023334  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
   (2)  023336  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
  6756  023340  000207                        RTS     PC              ;RETURN
  6757
  6758  023342  000000              6$:       .WORD   0               ;TRACK <CR> INDICATOR
  6759
  6760                              ;ROUTINE TO CONVERT INPUT DECIMAL ASCII NUMBERS TO BINARY
  6761                              ;       RETURN IF TERMINATOR (CR); RETURN +2 IF SEPARATOR (,).
  6762                              ;       R2 CONTAINS ADDRESS OF ASCII STRING
  6763
  6764  023344  005037  023470     CONVEA: CLR     HOLD            ;CLEAR WORKING LOCATION
  6765  023350  105712              1$:     TSTB    (R2)            ;SEE IF TERMINATOR
  6766  023352  001445                      BEQ     5$              ;BR IF TERMINATOR
  6767  023354  121237  023472             CMPB    (R2),SEPART     ;LOOK FOR THE SEPARATOR
  6768  023360  001437                      BEQ     4$              ;BR IF SEPARATOR
  6769  023362  121227  000060             CMPB    (R2),#'0        ;CHARATER GREATER OR EQUAL TO 0 ?
  6770  023366  103430                      BLO     3$              ;BR IF NOT
  6771  023370  121227  000072             CMPB    (R2),#':        ;IF CHARACTER GREATER THAN 9 ?
  6772  023374  103025                      BHIS    3$              ;BR IF IT IS
  6773  023376  005737  023470             TST     HOLD            ;VALUE IN WORKING LOCATION
  6774  023402  001412                      BEQ     2$              ;BR IF NONE
  6775  023404  006337  023470             ASL     HOLD            ;MULTIPLY 'HOLD' BY 2
  6776  023410  013746  023470             MOV     HOLD,-(SP)      ;THE THE '2 TIMES' VALUE
  6777  023414  006337  023470             ASL     HOLD            ;MULTIPLY 'HOLD' BY 2 AGAIN
  6778  023420  006337  023470             ASL     HOLD            ;MULTIPLY 'HOLD' BY 2 AGAIN
  6779  023424  062637  023470             ADD     (SP)+,HOLD      ;ADD THE '2 TIMES' VALUE
  6780  023430  005046              2$:     CLR     -(SP)           ;CLEAR THE STACK
  6781  023432  111216                      MOVB    (R2),(SP)       ;PUT THE CHARACTER ON THE STACK
  6782  023434  005202                      INC     R2              ;INCREMENT THE POINTER
  6783  023436  142716  000060             BICB    #60,(SP)        ;CLEAR THE UPPER BITS
  6784  023442  062637  023470             ADD     (SP)+,HOLD      ;ADD THE NEW CHARACTER
  6785  023446  000740                      BR      1$              ;GET THE NEXT CHARACTER
  6786  023450  012737  177777  023470     3$:     MOV     #-1,HOLD        ;SET BAD CHARACTER INDICATOR
  6787  023456  000403                      BR      5$              ;EXIT
  6788  023460  062705  000002             4$:     ADD     #2,R5           ;INCREMENT THE RETURN
  6789  023464  005202                      INC     R2              ;INCREMENT THE POINTER FOR SEPARATOR THEI
  6790  023466  000205              5$:     RTS     R5              ;RETURN
  6791
  6792  023470  000000              HOLD:   .WORD   0               ;WORKING LOCATION FOR THE CONVERSION
  6793  023472  000000              SEPART: .WORD                   ;SEPARATOR CHARACTER (IN ASCII) GOES HERE
  6794
  6795                              ;TYPEOUT ASSIGN/DEASSIGN ERROR MESSAGE
  6796
  6797  023474  142737  000007  001255 ASNERR: BICB    #7,UNIT+1       ;CLEAR PREVIOUS DRIVE NUMBER
  6798  023502  150437  001255             BISB    R4,UNIT+1       ;PRESENT DRIVE NUMBER
  6799  023506  104400  052241             TYPE    ,QUES           ;QUESTION MARK
  6800  023512  104400  051277             TYPE    ,UNTMSG         ;TYPE 'UNIT'
  6801  023516  104400  001254             TYPE    ,UNIT           ;UNIT NUMBER
  6802  023522  104400                      TYPE                    ;TYPE SPECIFIC MESSAGE
  6803  023524  000000              ASNMSG: .WORD   0               ;MESSAGE ADDRESS
  6804  023526  104400  001157             TYPE    ,$CRLF
  6805  023532  000207                      RTS     PC
  6806
  6807                              ;DEASSIGN A UNIT IF A FATAL ERROR OCCURS
```

```
 6808
 6809  023534  005004                    DROP:   CLR     R4              ;CLEAR R4 FOR UNIT NUMBER
 6810  023536  111004                            MOVB    (R0),R4         ;MOVE UNIT NUMBER TO R4
 6811  023540  105064  041550                    CLRB    ASNLST(R4)      ;REMOVE UNIT FROM ASSIGNED LIST
 6812  023544  006304                            ASL     R4              ;MAKE UNIT NUMBER INTO A TABLE INDEX
 6813  023546  010064  041560                    MOV     R0,DUNIT(R4)    ;PUT UNIT IN DROP LIST
 6814  023552  104400  001157                    TYPE    ,$CRLF
 6815  023556  104400  001157                    TYPE    ,$CRLF
 6816  023562  104400  051637                    TYPE    ,DROPNG         ;TYPE 'DROPPING UNIT'
 6817  023566  104400  051750                    TYPE    ,DRNUM          ;'DRIVE #'
 6818  023572  142737  000007  001255            BICB    #7,UNIT+1       ;CLEAR THE UNIT NUMBER
 6819  023600  151037  001255                    BISB    (R0),UNIT+1     ;SET THE UNIT NUMBER
 6820  023604  104400  001254                    TYPE    ,UNIT           ;TYPE THE UNIT NUMBER
 6821  023610  104400  001157                    TYPE    ,$CRLF
 6822  023614  000207                            RTS     PC
 6823
 6824                                    ;ROUTINE TO DEASSIGN A UNIT IF ERRORS BECOMES EXCESSIVE
 6825
 6826  023616  032737  000020  177570    ABNRML: BIT     #SW04,SWR       ;SEE IF SWITCH 4 SET
 6827  023624  001004                            BNE     1$              ;BR IF IT'S SET
 6828  023626  023760  001274  000062            CMP     MAXER,$TOTAL(R0)  ;CHECK TOTAL ERROR VALUE
 6829  023634  001737                            BEQ     DROP            ;DROP THE UNIT IF IT EXCEEDS MAX
 6830  023636  000207                    1$:     RTS     PC              ;RETURN
 6831
 6832                                    ;ROUTINE TO CHECK FOR END OF PASS AND END OF TEST
 6833
 6834  023640  005737  001314            NRML:   TST     ENDET           ;END OF PASS DETERMINED BY SEEKS OR WORDS ?
 6835  023644  001412                            BEQ     NRML1           ;BR IF SEEKS
 6836  023646  026037  000060  001262            CMP     $READ+2(R0),ENDCON+2  ;CHECK MSW OF WORDS READ COUNT
 6837  023654  101017                            BHI     NRML2           ;BR IF MSW GREATER THAN LIMIT
 6838  023656  103405                            BLO     NRML1           ;BR IF MSW LESS THAN LIMIT
 6839  023660  026037  000056  001260            CMP     $READ(R0),ENDCON  ;CHECK LSW AGAINST LIMIT
 6840  023666  103012                            BHIS    NRML2           ;BR IF EQUAL OR GREATER
 6841  023670  000506                            BR      NRMLX           ;EXIT
 6842  023672  026037  000050  001266    NRML1:  CMP     $POSIT+2(R0),ENDSEK+2  ;CHECK MSW OF SEEK COUNT
 6843  023700  101005                            BHI     NRML2           ;BR IF MSW GREATER THAN LIMIT
 6844  023702  103501                            BLO     NRMLX           ;EXIT IF MSW LESS THAN LIMIT
 6845  023704  026037  000046  001264            CMP     $POSIT(R0),ENDSEK  ;CHECK LSW OF SEEK COUNT
 6846  023712  103475                            BLO     NRMLX           ;EXIT IF LSW LESS THAN LIMIT
 6847  023714  104400  001157            NRML2:  TYPE    ,$CRLF          ;CR-LF
 6848  023720  104400  051673                    TYPE    ,ENDPAS         ;END OF PASS FOR THE UNIT
 6849  023724  016046  000074                    MOV     $PASSC(R0),-(SP)  ;PUT PASS COUNT ON THE STACK
 6850  023730  104410                            TYPOS                   ;CONVERT PASS COUNT TO DECIMAL AND TYPE IT
 6851  023732  142737  000007  001255            BICB    #7,UNIT+1       ;CLEAR THE UNIT NUMBER
 6852  023740  151037  001255                    BISB    (R0),UNIT+1     ;SET THE UNIT NUMBER
 6853  023744  032737  000020  177570            BIT     #SW04,SWR       ;SWITCH 4 SET ?
 6854  023752  001015                            BNE     1$              ;BR IF SET
 6855  023754  026037  000074  001270            CMP     $PASSC(R0),PASCNT  ;SEE IF AT END OF TEST
 6856  023762  103411                            BLO     1$              ;BR IF NOT
 6857  023764  104400  051707                    TYPE    ,ENDTST         ;TYPE 'END OF TEST'
 6858  023770  104400  051750                    TYPE    ,DRNUM          ;'DRIVE #'
 6859  023774  104400  001254                    TYPE    ,UNIT           ;DRIVE NUMBER
 6860  024000  104400  001157                    TYPE    ,$CRLF          ;CR-LF
 6861  024004  000427                            BR      3$              ;DEASSIGN THE DRIVE
```

```
                                                1$:     TYPE    ,DRNUM          ;'DRIVE #'
                                                        TYPE    ,UNIT           ;DRIVE NUMBER
                                                        TYPE    ,$CRLF          ;CR-LF
                                                        JSR     PC,TYPEST       ;TYPE THE UNIT'S STATISTICS
                                                        MOV     R3,-(SP)        ;SAVE R3
                                                        MOV     R4,-(SP)        ;SAVE R4
                                                        MOV     R0,R4           ;UNIT'S BLOCK ADDRESS
                                                        ADD     #$OPERC,R4      ;ADD THE STARTING ADDR OF SECTIONS TO CLEAR
                                                        MOV     #8.,R3          ;NUMBER OF LOCNS TO BE CLEARED
                                                                                ;(ERROR COUNTERS NOT CLEARED)
                                                2$:     CLR     (R4)+           ;CLEAR THE LOCN
                                                        DEC     R3              ;DECREMENT THE LOCATION COUNTER
                                                        BNE     2$              ;BR IF MORE TO GO
                                                        MOV     (SP)+,R4        ;RESTORE R4
                                                        MOV     (SP)+,R3        ;RESTORE R3
                                                        INC     $PASSC(R0)      ;INCREMENT THE PASS COUNT
                                                        BR      NRMLX           ;EXIT
                                                3$:     TYPE    ,$CRLF
                                                        CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
                                                        MOVB    (R0),R4         ;MOVE DRIVE NUMBER
                                                        CLRB    ASNLST(R4)      ;DELETE DRIVE FROM ASSIGNED LIST
                                                        ASL     R4              ;MAKE DRIVE NUMBER INTO TABLE INDEX
                                                        MOV     R0,DUNIT(R4)    ;PUT BLOCK ADDRESS INTO DROP LIST
                                                NRMLX:  RTS     PC              ;RETURN

                                                ;ROUTINE TO GET THE REMAINDER OF THE RANDOM NUMBER

                                                GETREM: MOV     $LONUM,-(SP)    ;STORE RANDOM NUMBER ON THE STACK FOR DIVIDE
                                                        MOV     $HINUM,-(SP)    ;UPPER PART
                                                        MOV     R5,-(SP)        ;PUT THE DIVISOR ONTO THE STACK
                                                        JSR     PC,LINKDV       ;DIVIDE THE RANDOM NUMBERS
                                                        MOV     (SP)+,R5        ;PUT THE REMAINDER INTO R5
                                                        TST     (SP)+           ;ADJUST THE STACK POINTER
                                                        NOP                     ;FOR DEBUGGING HALT
                                                        RTS     PC

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

                                                LINKDV:
                                                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                                                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
                                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                                        MOV     16(SP),R5       ;DIVISOR
                                                        CLR     R4              ;OTHER DIVISOR WORD
                                                        MOV     20(SP),R2       ;UPPER DIVIDEND WORD
                                                        MOV     22(SP),R3       ;LOWER DIVIDEND WORD
                                                        CLR     R0              ;CLEAR OTHER DIVIDEND REGISTERS
                                                        CLR     R1
                                                        JSR     PC,M.DPID       ;GO TO THE DIVIDE ROUTINE
```

```
6910  024200  010166  000020                MOV     R1,20(SP)       ;REMAINDER ON THE STACK
6911  024204  010366  000022                MOV     R3,22(SP)       ;QUOTIENT ON THE STACK
6912  024210  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
  (2) 024212  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2) 024214  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2) 024216  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2) 024220  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2) 024222  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
6913  024224  012616                        MOV     (SP)+,(SP)      ;MOVE RETURN UP THE STACK
6914  024226  000207                        RTS     PC
6915
6916                                ;       DIVISION UTILITY SUBROUTINE
6917                                ;       R0-R1-R2-R3=DIVIDEND
6918                                ;       R4-R5=DIVISOR
6919                                ;       R0-R1=REMAINDER AFTER DIVISION
6920                                ;       R2-R3=QUOTIENT AFTER DIVISION
6921                                ;       ENTER WITH JSR  PC,M.DPID
6922                                ;
6923  024230  012746  000040       M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
6924  024234  010446                        MOV     R4,-(SP)        ;HIGH ORDER
6925  024236  010546                        MOV     R5,-(SP)        ;LOW ORDER DIVISOR TO THE STACK
6926  024240  005466  000002                NEG     2(SP)           ;FORM NEGATIVE
6927  024244  005416                        NEG     @SP             ;VERSION OF THE DIVISOR
6928  024246  005666  000002                SBC     2(SP)
6929  024252  061601                        ADD     @SP,R1
6930  024254  005500                        ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
6931  024256  066600  000002                ADD     2(SP),R0
6932  024262  103445                        BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
6933  024264  005046                        CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
6934  024266  006103                M.DP40: ROL     R3
6935  024270  006102                        ROL     R2
6936  024272  006101                        ROL     R1
6937  024274  006100                        ROL     R0
6938  024276  005716                        TST     @SP             ;TEST "CARRY" INDICATOR
6939  024300  001410                        BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
6940  024302  005016                        CLR     @SP             ;CLEAR UP FOR NEXT TIME
6941  024304  066601  000002                ADD     2(SP),R1
6942  024310  005500                        ADC     R0              ;ADD -(DIVISOR)
6943  024312  005516                        ADC     @SP     ;  I    ;SET "CARRY"
6944  024314  066600  000004                ADD     4(SP),R0;<-
6945  024320  000404                        BR      M.DP42
6946  024322  060501                M.DP41: ADD     R5,R1
6947  024324  005500                        ADC     R0              ;ADD +(DIVISOR)
6948  024326  005516                        ADC     @SP     ;  I    ;SET "CARRY"
6949  024330  060400                        ADD     R4,R0   ;<-
6950  024332  005516                M.DP42: ADC     @SP             ;SET "CARRY"
6951  024334  005716                        TST     @SP             ;TEST THE UPDATE INDICATOR
6952  024336  001401                        BEQ     .+4     ;->     ;IF ZERO FORGET IT
6953  024340  005203                        INC     R3      ;  I    ;NO CARRY POSSIBLE HERE
6954  024342  005366  000006                DEC     6(SP)   ;<-     ;DECREMENT COUNTER
6955  024346  003347                        BGT     M.DP40          ;BRANCH IF MORE TO DO
6956  024350  006003                        ROR     R3
6957  024352  103404                        BCS     M.DP44
6958  024354  060501                        ADD     R5,R1
```

```
 6959  024356  005500                          ADC     R0
 6960  024360  060400                          ADD     R4,R0
 6961  024362  000241                          CLC
 6962  024364  006103                  M.DP44: ROL     R3
 6963  024366  062706  000010                  ADD     #10,SP          ;ADJUST STACK BY 4 WORDS
 6964  024372  000242                          CLV
 6965  024374  000207                          RTS     PC
 6966  024376  062706  000006          M.DP50: ADD     #6,SP
 6967  024402  000262                          SEV
 6968  024404  000207                          RTS     PC
 6969
 6970
 6971
 6972                                  ;LINK SUBROUTINE TO MULTIPLY ROUTINE 'M.DPIM'
 6973
 6974  024406                          LINKML:
  (2)  024406  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  024410  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  024412  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  024414  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  024416  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  024420  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 6975  024422  005002                          CLR     R2              ;PUT MULTIPLIER IN R2 & R3
 6976  024424  016603  000020                  MOV     20(SP),R3
 6977  024430  005004                          CLR     R4              ;PUT MULTIPLICAND IN R4 & R5
 6978  024432  016605  000016                  MOV     16(SP),R5
 6979  024436  004737  024470                  JSR     PC,M.DPIM       ;MULTIPLY R0 & R1 BY R2 & R3
 6980  024442  010266  000016                  MOV     R2,16(SP)       ;HIGH ORDER PRODUCT
 6981  024446  010366  000020                  MOV     R3,20(SP)       ;LOW ORDER PRODUCT
 6982  024452  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
  (2)  024454  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2)  024456  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)  024460  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  024462  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  024464  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 6983  024466  000207                          RTS     PC
 6984
 6985                                  ;       SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
 6986                                  ;       USES ALL REGISTERS (R0-R5)
 6987                                  ;
 6988                                  ;       ENTER WITH JSR  PC,M.DPIM
 6989                                  ;               MULTIPLIER IN R2-R3
 6990                                  ;               MULTIPLICAND IN R4-R5
 6991                                  ;               RESULT RETURNED IN R0-R1-R2-R3
 6992                                  ;
 6993  024470  005000                  M.DPIM: CLR     R0              ;CLEAR HIGH ORDER WORDS
 6994  024472  005001                          CLR     R1
 6995  024474  012746  000041                  MOV     #41,-(SP)       ;MOVE 33 DEC TO COUNTER
 6996  024500  006000                  M.DP01: ROR     R0
 6997  024502  006001                          ROR     R1
 6998  024504  006002                          ROR     R2              ;SHIFT TO ADD
 6999  024506  006003                          ROR     R3
 7000  024510  103003                          BCC     M.DP02          ;NO CARRY NO ADD
 7001  024512  060501                          ADD     R5,R1
```

```
7002  024514  005500                          ADC     R0              ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
7003  024516  060400                          ADD     R4,R0           ;PRODUCT
7004  024520  005316                M.DP02:  DEC     @SP             ;DECREMENT COUNTER
7005  024522  001366                          BNE     M.DP01
7006  024524  005726                          TST     (SP)+           ;REMOVE THE COUNTER
7007  024526  000207                          RTS     PC
7008
7009                                  ;ROUTINE TO REPLACE LEADING ZEROS IN A NUMERIC STRING WITH SPACES
7010
7011  024530  012737  000004  024650  $RPZR8:  MOV     #4,RPZRC        ;SETUP TO TYPE 8 SIGNIFICANT DIGITS
7012  024536  000412                          BR      RPZERO          ;TYPE THE VALUE
7013  024540  012737  000006  024650  $RPZR4:  MOV     #6,RPZRC        ;SETUP TO TYPE 4 SIGNIFICANT DIGITS
7014  024546  000406                          BR      RPZERO          ;TYPE THE VALUE
7015  024550  012737  000007  024650  $RPZR5:  MOV     #7,RPZRC        ;SETUP TO TYPE 3 SIGNIFICANT DIGITS
7016  024556  000402                          BR      RPZERO          ;TYPE THE VALUE
7017  024560  005037  024650          $RPZR0:  CLR     RPZRC           ;SETUP TO TYPE 10 SIGNIFICANT DIGITS
7018  024564  010046                RPZERO:  MOV     R0,-(SP)        ;SAVE R0
7019  024566  016600  000004                  MOV     4(SP),R0        ;ADDRESS OF NUMBER TO R0
7020  024572  122710  000060          1$:     CMPB    #'0,(R0)        ;BYTE EQUAL TO ASCII '0' ?
7021  024576  001004                          BNE     2$              ;BR IF NOT
7022  024600  112710  000040                  MOVB    #40,(R0)        ;REPLACE THE ZERO WITH A SPACE
7023  024604  005200                          INC     R0              ;INCREMENT THE BYTE ADDRESS
7024  024606  000771                          BR      1$              ;GO BACK AND LOOK FOR MORE LEADING ZEROS
7025  024610  105710                2$:     TSTB    (R0)            ;SEE IF ZERO BYTE TERMINATOR
7026  024612  001003                          BNE     3$              ;BR IF NOT
7027  024614  005300                          DEC     R0              ;BACKUP STRING POINTER
7028  024616  112710  000060                  MOVB    #'0,(R0)        ;PUT A ZERO BACK IN
7029  024622  016637  000004  024640  3$:     MOV     4(SP),4$        ;PUT ADDRESS IN LOCATION FOR TYPEOUT
7030  024630  063737  024650  024640          ADD     RPZRC,4$        ;BEGINNING OF SIGNIFICANT DIGITS
7031  024636  104400                          TYPE                    ;TYPE THE NUMBER
7032  024640  000000                4$:     .WORD   0               ;ADDRESS OF NUMBER
7033  024642  012600                          MOV     (SP)+,R0        ;RESTORE R0
7034  024644  012616                          MOV     (SP)+,(SP)      ;MOVE RETURN ADDRESS
7035  024646  000207                          RTS     PC              ;RETURN
7036
7037  024650  000000                RPZRC:   .WORD   0               ;OFFSET FOR CONVERTED DIGITS HERE
7038
7039                                  ;CONVERT AN OCTAL NUMBER TO ASCII AND PRINT IT
7040                                  ;       (ROUTINE ADAPTED FROM '$TYPOC')
7041
7042  024652  112737  000001  025043  PRTOCT:  MOVB    #1,FILLO        ;SET THE ZERO FILL SWITCH
7043  024660  112737  000006  025045          MOVB    #6,OMODE+1      ;SET FOR SIX(6) DIGITS
7044  024666  112737  000005  025042          MOVB    #5,OCNT         ;SET THE ITERATION COUNT
7045  024674  010346                          MOV     R3,-(SP)        ;SAVE R3
7046  024676  010446                          MOV     R4,-(SP)        ;SAVE R4
7047  024700  010546                          MOV     R5,-(SP)        ;SAVE R5
7048  024702  113704  025045                  MOVB    OMODE+1,R4      ;GET THE NUMBER OF DIGITS TO TYPE
7049  024706  005404                          NEG     R4
7050  024710  062704  000006                  ADD     #6,R4           ;SUBTRACT IT FOR MAX. ALLOWED
7051  024714  110437  025044                  MOVB    R4,OMODE        ;SAVE IT FOR USE
7052  024720  113704  025043                  MOVB    FILLO,R4        ;GET THE ZERO FILL SWITCH
7053  024724  016605  000010                  MOV     10(SP),R5       ;PICKUP THE INPUT NUMBER
7054  024730  005003                          CLR     R3              ;CLEAR THE OUTPUT WORD
7055  024732  006105                1$:     ROL     R5              ;ROTATE MSB INTO "C"
```

```
7056  024734  000404                        BR      3$              ;GO DO MSB
7057  024736  006105              2$:       ROL     R5              ;FORM THIS DIGIT
7058  024740  006105                        ROL     R5
7059  024742  006105                        ROL     R5
7060  024744  010503                        MOV     R5,R3
7061  024746  006103              3$:       ROL     R3              ;GET LSB OF THIS DIGIT
7062  024750  105337  025044                DECB    OMODE           ;TYPE THIS DIGIT?
7063  024754  100016                        BPL     7$              ;BR IF NO
7064  024756  042703  177770                BIC     #177770,R3      ;GET RID OF JUNK
7065  024762  001002                        BNE     4$              ;TEST FOR 0
7066  024764  005704                        TST     R4              ;SUPPRESS THIS 0?
7067  024766  001403                        BEQ     5$              ;BR IF YES
7068  024770  005204              4$:       INC     R4              ;DON'T SUPPRESS ANYMORE 0'S
7069  024772  052703  000060                BIS     #'0,R3          ;MAKE THIS DIGIT ASCII
7070  024776  052703  000040    5$:       BIS     #' ,R3          ;MAKE ASCII IF NOT ALREADY
7071  025002  110337  025040                MOVB    R3,8$           ;SAVE FOR TYPING
7072  025006  104422  025040                DISPLY  ,8$             ;GO PRINT THIS DIGIT
7073  025012  105337  025042    7$:       DECB    OCNT            ;COUNT BY 1
7074  025016  003347                        BGT     2$              ;BR IF MORE TO DO
7075  025020  002402                        BLT     6$              ;BR IF DONE
7076  025022  005204                        INC     R4              ;INSURE LAST DIGIT ISN'T A BLANK
7077  025024  000744                        BR      2$              ;GO DO THE LAST DIGIT
7078  025026  012605              6$:       MOV     (SP)+,R5        ;RESTORE R5
7079  025030  012604                        MOV     (SP)+,R4        ;RESTORE R4
7080  025032  012603                        MOV     (SP)+,R3        ;RESTORE R3
7081  025034  012616                        MOV     (SP)+,(SP)      ;SET UP THE STACK FOR RETURNING
7082  025036  000207                        RTS     PC              ;RETURN
7083  025040     000              8$:       .BYTE   0               ;STORAGE FOR ASCII DIGIT
7084  025041     000                        .BYTE   0               ;TERMINATOR FOR TYPE ROUTINE
7085  025042     000              OCNT:     .BYTE   0               ;OCTAL DIGIT COUNTER
7086  025043     000              FILLO:    .BYTE   0               ;ZERO FILL SWITCH
7087  025044  000000              OMODE:    0                       ;NUMBER OF DIGITS TO TYPE
7088
7089                              ;ROUTINE TO TYPE AT PRIORITY 4
7090
7091  025046  013746  177776      TYPRI4:   MOV     @#PS,-(SP)      ;SAVE THE PRESENT STATUS
7092  025052  012737  000200  177776        MOV     #200,@#PS       ;CHANGE THE PRIORITY TO 4
7093  025060  012537  025070                MOV     (R5)+,1$        ;MESSAGE ADDRESS
7094  025064  004737  025550                JSR     PC,$TYPE        ;TYPE THE MESSAGE
7095  025070  000000              1$:       .WORD   0               ;MESSAGE ADDRESS GOES HERE
7096  025072  000205                        RTS     R5              ;RETURN
7097
7098                              ;ROUTINE TO ROUTE MESSAGES TO EITHER THE PRINTER (IF AVAILABLE) OR
7099                              ;       THE TELETYPE
7100                              ;NOTE:  $DSPLY MUST BE DEFINED BY 'SETTRAP'
7101
7102  025074  032737  020000  177570  $DSPLY:   BIT     #SW13,SWR       ;INHIBIT TYPEOUTS
7103  025102  001027                        BNE     5$              ;BR IF SET
7104  025104  012737  000200  177776  1$:       MOV     #200,@#PS       ;CHANGE THE PRIORITY TO 4
7105  025112  005737  001204                TST     $PRFLG          ;PRINTER ON THE SYSTEM ?
7106  025116  100411                        BMI     3$              ;BR IF NOT
7107  025120  017637  000000  025136        MOV     @(SP),2$        ;GET THE MESSAGE ADDRESS
7108  025126  013746  177776                MOV     @#PS,-(SP)      ;PUT THE PROC STATUS ON THE STACK
7109  025132  004737  025656                JSR     PC,$PRINT       ;PRINT THE MESSAGE
```

```
7110  025136  000000                    2$:     .WORD   0               ;MESSAGE ADDRESS GOES HERE
7111  025140  000410                            BR      5$              ;EXIT
7112  025142  017637  000000  025160    3$:     MOV     @(SP),4$        ;SET UP TO TYPE THE MESSAGE
7113  025150  013746  177776                    MOV     @#PS,-(SP)      ;PUT THE PROC STATUS ON THE STACK
7114  025154  004737  025550                    JSR     PC,$TYPE        ;TYPE THE MESSAGE
7115  025160  000000                    4$:     .WORD   0               ;MESSAGE ADDRESS GOES HERE
7116  025162  062716  000002            5$:     ADD     #2,(SP)         ;RETURN ADDRESS
7117  025166  000002                            RTI
7118
7119                                    ;SAVE THE REGISTERS IF A DRIVER ERROR CALL
7120
7121  025170  010537  001252            ERENTR: MOV     R5,ATTN         ;SAVE THE ATTENTION REGISTER CONTENTS
7122  025174  010137  001250                    MOV     R1,DRIVE        ;DRIVE NUMBER
7123  025200  032737  020000  177570            BIT     #SW13,SWR       ;INHIBIT PRINTOUTS ?
7124  025206  001002                            BNE     1$              ;BR IF YES
7125  025210  004737  021314                    JSR     PC,$TIME        ;TYPE THE TIME
7126  025214  000207                    1$:     RTS     PC
7127
7128                                    ;ROUTINE TO GET AN OCTAL PARAMETER
7129                                            ;RETURN +2 IF CARRIAGE RETURN ENTERED
7130
7131  025216  011646                    GETOCT: MOV     (SP),-(SP)      ;PROVIDE SPACE FOR THE
7132  025220  104414                    1$:     RDLIN                   ;READ AN ASCIZ LINE
7133  025222  012600                            MOV     (SP)+,R0        ;GET ADDRESS OF 1ST CHARACTER
7134  025224  010037  025330                    MOV     R0,5$           ;AND SAVE IT
7135  025230  105710                            TSTB    (R0)            ;FIRST CHARACTER A 'CR'
7136  025232  001442                            BEQ     6$              ;BR IF IT IS
7137  025234  122710  000003                    CMPB    #3,(R0)         ;CONTROL C ?
7138  025240  001443                            BEQ     8$              ;BR IF IT IS
7139  025242  005001                            CLR     R1              ;CLEAR DATA WORD
7140  025244  005002                            CLR     R2
7141  025246  112046                    2$:     MOVB    (R0)+,-(SP)     ;PICKUP THIS CHARACTER
7142  025250  001420                            BEQ     3$              ;IF ZERO GET OUT
7143  025252  122716  000060                    CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER
7144  025256  003021                            BGT     4$              ;IS AN OCTAL DIGIT
7145  025260  122716  000067                    CMPB    #'7,(SP)
7146  025264  002416                            BLT     4$
7147  025266  006301                            ASL     R1              ;*2
7148  025270  006102                            ROL     R2
7149  025272  006301                            ASL     R1              ;*4
7150  025274  006102                            ROL     R2
7151  025276  006301                            ASL     R1              ;*8
7152  025300  006102                            ROL     R2
7153  025302  042716  177770                    BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
7154  025306  062601                            ADD     (SP)+,R1        ;ADD IN THIS DIGIT
7155  025310  000756                            BR      2$              ;LOOP
7156  025312  005726                    3$:     TST     (SP)+           ;CLEAN TERMINATOR FROM STACK
7157  025314  010166  000002                    MOV     R1,2(SP)        ;SAVE THE RESULT
7158  025320  000412                            BR      7$              ;EXIT
7159  025322  005726                    4$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
7160  025324  105010                            CLRB    (R0)            ;SET A TERMINATOR
7161  025326  104400                            TYPE                    ;TYPE UP THRU THE BAD CHAR.
7162  025330  000000                    5$:     .WORD   0
7163  025332  104400  001156                    TYPE    ,$QUES          ;"?" "CR" & "LF"
```

```
7164  025336  000730                        BR      1$              ;TRY AGAIN
7165  025340  012616              6$:       MOV     (SP)+,(SP)      ;RESTORE THE PC
7166  025342  062716  000002                ADD     #2,(SP)         ;INCREMENT RETURN
7167  025346  000207              7$:       RTS     PC              ;RETURN
7168  025350  000137  002162      8$:       JMP     ENTPR           ;CONTROL C ENTERED
7169
7170                              ;ROUTINE TO GET DECIMAL PARAMETERS
7171                                      ;RETURN +2 IF CARRIAGE RETURN ENTERED
7172
7173  025354  011646              GETDEC:   MOV     (SP),-(SP)      ;PROVIDE SPACE FOR
7174  025356  104414              1$:       RDLIN                   ;READ AN ASCIZ LINE
7175  025360  012600                        MOV     (SP)+,R0        ;ADDRESS OF 1ST CHAR.
7176  025362  010037  025464                MOV     R0,6$           ;SAVE IN CASE OF BAD INPUT
7177  025366  105710                        TSTB    (R0)            ;FIRST CHARACTER A 'CR' ?
7178  025370  001441                        BEQ     7$              ;BR IF IT IS
7179  025372  122710  000003                CMPB    #3,(R0)         ;CONTROL C ?
7180  025376  001442                        BEQ     9$              ;BR IF IT IS
7181  025400  005046                        CLR     -(SP)           ;CLEAR DATA WORD
7182  025402  112001              2$:       MOVB    (R0)+,R1        ;PICKUP THIS CHARACTER
7183  025404  001421                        BEQ     4$              ;GET OUT IF ZERO
7184  025406  122701  000060                CMPB    #'0,R1          ;MAKE SURE THIS CHARACTER
7185  025412  003021                        BGT     5$              ;IS A DIGIT BETWEEN 0 & 9
7186  025414  122701  000071                CMPB    #'9,R1
7187  025420  002416                        BLT     5$
7188  025422  006316                        ASL     (SP)            ;*2
7189  025424  011646                        MOV     (SP),-(SP)      ;SAVE FOR LATER
7190  025426  006316                        ASL     (SP)            ;*4
7191  025430  006316                        ASL     (SP)            ;*8.
7192  025432  062616                        ADD     (SP)+,(SP)      ;*10.
7193  025434  102410                        BVS     5$              ;OVERFLOW ISN'T ALLOWED
7194  025436  162701  000060                SUB     #'0,R1          ;STRIP AWAY THE ASCII JUNK
7195  025442  060116                        ADD     R1,(SP)         ;ADD IN THIS DIGIT
7196  025444  102404                        BVS     5$              ;OVERFLOW ISN'T ALLOWED
7197  025446  000755                        BR      2$              ;LOOP
7198  025450  012666  000002      4$:       MOV     (SP)+,2(SP)     ;SAVE THE RESULT
7199  025454  000412                        BR      8$              ;EXIT
7200  025456  005726              5$:       TST     (SP)+           ;CLEAN PARTIAL NUMBER FROM STACK
7201  025460  105010                        CLRB    (R0)            ;SET A TERMINATOR
7202  025462  104400                        TYPE                    ;TYPE THE INPUT UP TO BAD CHAR.
7203  025464  000000              6$:       .WORD   0               ;POINTER GOES HERE
7204  025466  104400  001156                TYPE    ,$QUES          ;"?" "CR" &"LF"
7205  025472  000731                        BR      1$              ;TRY AGAIN
7206  025474  012616              7$:       MOV     (SP)+,(SP)      ;RESTORE THE PC
7207  025476  062716  000002                ADD     #2,(SP)         ;INCREMENT THE RETURN
7208  025502  000207              8$:       RTS     PC              ;RETURN
7209  025504  000137  002162      9$:       JMP     ENTPR           ;CONTROL C ENTERED
7210
7211                              ;*****************************************************************
7212
7213                              .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
7214
7215                              ;*CALL
7216                              ;*      MOV     #NUMADR,-(SP)           ;FIRST ADDRESS OF ASCIZ STRING
7217                              ;*      JSR     PC,@#$SUPRS
```

```
7218                                    ;
7219                                    ;NOTE:  ROUTINE MODIFIED FROM THE 'SYSMAC' ROUTINE
7220
7221
7222  025510  010046                    $SUPRS: MOV     R0,-(SP)            ;SAVE R0
7223  025512  016600  000004                    MOV     4(SP),R0            ;PICKUP THE POINTER
7224  025516  105710                    1$:     TSTB    (R0)                ;TERMINATEOR?
7225  025520  001403                            BEQ     2$                  ;BR IF YES
7226  025522  122720  000060                    CMPB    #'0,(R0)+           ;IS THIS AN ASCII "0" ?
7227  025526  001773                            BEQ     1$                  ;BR IF YES
7228  025530  005300                    2$:     DEC     R0                  ;BACKUP BY "1"
7229  025532  010037  025540                    MOV     R0,3$               ;SAVE FOR TYPING
7230  025536  104422                            DISPLY                      ;GO PRINT
7231  025540  000000                    3$:     .WORD   0                   ;ASCIZ POINTER GOES HERE
7232  025542  012600                            MOV     (SP)+,R0            ;RESTORE R0
7233  025544  012616                            MOV     (SP)+,(SP)          ;RESTORE THE STACK
7234  025546  000207                            RTS     PC                  ;RETURN
7235
7236                                    ;******************************************************************
7237
7238                                    .SBTTL  TYPE ROUTINE
7239
7240                                    ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7241                                    ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7242                                    ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7243                                    ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7244                                    ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7245                                    ;*
7246                                    ;*CALL:
7247                                    ;*1) USING A TRAP INSTRUCTION
7248                                    ;*      TYPE    ,MESADR             ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7249                                    ;*OR
7250                                    ;*      TYPE
7251                                    ;*      MESADR
7252                                    ;*
7253                                    ;*2) USING A JSR INSTRUCTION
7254                                    ;*      MOV     PS,-(SP)            ;PUSH PROCESSOR STATUS WORD ON THE STACK
7255                                    ;*      JSR     PC,$TYPE            ;CALL TYPE ROUTINE
7256                                    ;*      MESADDR                     ;FIRST ADRESS OF MESSAGE
7257
7258  025550  105737  001151            $TYPE:  TSTB    $TPFLG              ;IS THERE A TERMINAL?
7259  025554  100002                            BPL     1$                  ;BR IF YES
7260  025556  000000                            HALT                        ;HALT HERE IF NO TERMINAL
7261  025560  000407                            BR      3$                  ;LEAVE
7262  025562  010046                    1$:     MOV     R0,-(SP)            ;SAVE R0
7263  025564  017600  000002                    MOV     @2(SP),R0           ;GET ADDRESS OF ASCIZ STRING
7264  025570  112046                    2$:     MOVB    (R0)+,-(SP)         ;PUSH CHARACTER TO BE TYPED ONTO STACK
7265  025572  001005                            BNE     4$                  ;BR IF IT ISN'T THE TERMINATOR
7266  025574  005726                            TST     (SP)+               ;IF TERMINATOR POP IT OFF THE STACK
7267  025576  012600                            MOV     (SP)+,R0            ;RESTORE R0
7268  025600  062716  000002            3$:     ADD     #2,(SP)             ;ADJUST RETURN PC
7269  025604  000002                            RTI                         ;RETURN
7270  025606  004737  025640            4$:     JSR     PC,7$               ;GO TYPE THIS CHARACTER
7271  025612  123726  001150            5$:     CMPB    $FILLC,(SP)+        ;IS IT TIME FOR FILLER CHARS.?
```

```
7272  025616  001364                          BNE     2$                      ;IF NO GO GET NEXT CHAR.
7273  025620  013746  001146                  MOV     $NULL,-(SP)             ;GET # OF FILLER CHARS. NEEDED
7274                                                                          ;AND THE NULL CHAR.
7275  025624  105366  000001          6$:     DECB    1(SP)                   ;DOES A NULL NEED TO BE TYPED?
7276  025630  002770                          BLT     5$                      ;BR IF NO--GO POP THE NULL OFF OF STACK
7277  025632  004737  025640                  JSR     PC,7$                   ;GO TYPE A NULL
7278  025636  000772                          BR      6$                      ;LOOP
7279  025640  105777  153276          7$:     TSTB    @$TPS                   ;WAIT UNTIL PRINTER IS READY
7280  025644  100375                          BPL     7$
7281  025646  116677  000002  153270          MOVB    2(SP),@$TPB             ;LOAD CHAR TO BE TYPED INTO DATA REG.
7282  025654  000207                          RTS     PC
7283
7284                                  ;*****************************************************************
7285
7286                                  .SBTTL  PRINT ROUTINE
7287
7288                                  ;*ROUTINE TO PRINT ASCIZ MESSAGE.  MESSAGE MUST TERMINATE WITH A 0 BYTE
7289                                  ;*
7290                                  ;*CALL:
7291                                  ;*1) USING A TRAP INSTRUCTION
7292                                  ;*      DISPLY  ,MESADR         ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7293                                  ;*
7294                                  ;*OR
7295                                  ;*      PRINT
7296                                  ;*      MESADR
7297                                  ;*
7298                                  ;*2) USING A JSR INSTRUCTION
7299                                  ;*      MOV     PS,-(SP)                ;PUSH PROCESSOR STATUS WORD ON THE STACK
7300                                  ;*      JSR     PC,$PRINT               ;CALL PRINT ROUTINE
7301                                  ;*      MESADDR                         ;FIRST ADRESS OF MESSAGE
7302
7303  025656  005737  001204          $PRINT: TST     $PRFLG          ;IS THERE A PRINTER
7304  025662  100002                          BPL     1$              ;BR IF YES
7305  025664  000000                          HALT
7306  025666  000412                          BR      7$              ;RETURN & TYPE INSTEAD
7307  025670  005777  153312          1$:     TST     @$LSCS          ;PRINTER ERROR ?
7308  025674  100012                          BPL     6$              ;BR IF NO ERROR
7309  025676  005737  001212                  TST     $PRTER          ;WAS ERROR SET BEFORE ?
7310  025702  001004                          BNE     7$              ;BR IF YES, EXIT & TYPE INSTEAD
7311  025704  005137  001212                  COM     $PRTER          ;SET PRINTER ERROR FLAG
7312  025710  104400  052243                  TYPE    ,PRTATN         ;TYPE 'PRINTER ATTENTION'
7313  025714  062716  000004          7$:     ADD     #4,(SP)         ;INCREMENT RETURN TO GO TO TYPE ROUTINE
7314  025720  000002                          RTI                     ;RETURN
7315  025722  005037  001212          6$:     CLR     $PRTER          ;CLEAR PRINTER ERROR FLAG
7316  025726  010046                          MOV     R0,-(SP)        ;SAVE R0
7317  025730  017600  000002                  MOV     @2(SP),R0       ;GET ADDRESS OF ASCIZ STRING
7318  025734  005746                          TST     -(SP)           ;MOVE THE STACK POINTER DOWN
7319  025736  112016          2$:     MOVB    (R0)+,(SP)      ;PUSH CHARACTER TO BE TYPED ONTO THE STACK.
7320  025740  001005                          BNE     4$              ;BR IF IT ISN'T THE TERMINATOR
7321  025742  005726                          TST     (SP)+           ;IF TERMINATOR, POP IT OFF THE STACK
7322  025744  012600                          MOV     (SP)+,R0        ;RESTORE R0
7323  025746  062716  000002          3$:     ADD     #2,(SP)         ;ADJUST RETURN PC
7324  025752  000002                          RTI                     ;RETURN
7325  025754  004737  025762          4$:     JSR     PC,5$           ;PRINT THIS CHARACTER
```

```
 7326  025760  000766                        BR      2$                      ;GET NEXT CHAR
 7327  025762  105777  153220          5$:   TSTB    @$LSCS                  ;WAIT UNTIL PRINTER READY
 7328  025766  100375                        BPL     5$
 7329  025770  116677  000002  153212        MOVB    2(SP),@$LSDB            ;LOAD CHAR TO BE PRINTED INTO DATA REG
 7330  025776  000207                        RTS     PC
 7331
 7332                          ;*****************************************************************
 7333
 7334                          .SBTTL  TTY INPUT ROUTINE
 7335
 7336                          ;*TK INITIALIZE ROUTINE
 7337                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 7338                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 7339                          ;
 7340                          ;*CALL
 7341                          ;*      JSR     PC,$TKINT
 7342                          ;*      RETURN
 7343                          ;
 7344  026000  005037  026172  $TKINT: CLR     $TKCNT                  ;CLEAR COUNT OF ITEMS IN QUEUE
 7345  026004  012737  026176  026174        MOV     #$TKQSRT,$TKQIN         ;MOVE THE STARTING ADDRESS
 7346  026012  012737  026042  000060        MOV     #$TKSRV,@#TKVEC         ;INITIALIZE THE KEYBOARD VECTOR
 7347  026020  012737  000200  000062        MOV     #200,@#TKVEC+2          ;"BR" LEVEL 4
 7348  026026  005777  153106                TST     @$TKB                   ;CLEAR DONE FLAG
 7349  026032  012777  000100  153076        MOV     #BIT06,@$TKS            ;ENABLE INTERRUPT
 7350  026040  000207                        RTS     PC                      ;RETURN TO CALLER
 7351
 7352                          ;*TK SERVICE ROUTINE
 7353                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 7354                          ;
 7355  026042  117746  153072  $TKSRV: MOVB    @$TKB,-(SP)             ;PICKUP THE CHARACTER
 7356  026046  042716  177600        BIC     #^C177,(SP)             ;STRIP THE JUNK
 7357  026052  005737  026172        TST     $TKCNT                  ;SEE IF FIRST CHARACTER
 7358  026056  001004                BNE     1$                      ;BR IF NOT
 7359  026060  004737  021314        JSR     PC,$TIME                ;TYPE THE TIME
 7360  026064  104400  001157        TYPE    ,$CRLF                  ;CR-LF
 7361  026070  111637  026170  1$:   MOVB    (SP),5$                 ;ECHO THE CHARACTER
 7362  026074  104400  026170        TYPE    ,5$
 7363  026100  022716  000015        CMP     #15,(SP)                ;CARRIAGE RETURN ?
 7364  026104  001416                BEQ     3$                      ;BR IF CR
 7365  026106  023727  026174  026200  CMP     $TKQIN,#$TKQEND         ;AT THE END ?
 7366  026114  001003                BNE     2$                      ;BR IF NOT AT END
 7367  026116  104400  001152        TYPE    ,$BELL                  ;RING THE TTY BELL
 7368  026122  000420                BR      4$                      ;EXIT
 7369  026124  111677  000044  2$:   MOVB    (SP),@$TKQIN            ;PUT IN QUEUE
 7370  026130  005237  026172        INC     $TKCNT                  ;COUNT THE CHARACTER
 7371  026134  005237  026174        INC     $TKQIN                  ;UPDATE THE POINTER
 7372  026140  000411                BR      4$                      ;EXIT
 7373  026142  004737  021634  3$:   JSR     PC,KSR                  ;PROCESS THE ENTRY
 7374  026146  005037  026172        CLR     $TKCNT                  ;CLEAR THE CHARACTER COUNTER
 7375  026152  012737  026176  026174  MOV     #$TKQSRT,$TKQIN         ;RESET THE POINTER
 7376  026160  005037  026176        CLR     $TKQSRT                 ;CLEAR THE BUFFER
 7377  026164  005726          4$:   TST     (SP)+                   ;RESET THE STACK POINTER
 7378  026166  000002                RTI                             ;RETURN
 7379  026170     000          5$:   .BYTE   0                       ;STORAGE FOR ASCII CHARACTER TO TYPE
```

```
 7380  026171  000                    .BYTE   0                       ;TERMINATOR
 7381  026172  000000         $TKCNT: .WORD   0                       ;NUMBER OF ITEMS IN THE QUEUE
 7382  026174  000000         $TKQIN: .WORD   0                       ;INPUT POINTER
 7383  026176  000002         $TKQSRT: .BLKB  2                       ;TTY KEYBOARD QUEUE
 7384          026200         $TKQEND =       .
 7385                         .EVEN
 7386
 7387                         ;*****************************************************************
 7388                         ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 7389                         ;*CALL:
 7390                         ;*      RDCHR                   ;INPUT A SINGLE CHARACTER FROM THE TTY
 7391                         ;*      RETURN HERE             ;CHARACTER IS ON THE STACK
 7392                         ;
 7393
 7394  026200  011646         $RDCHR: MOV     (SP),-(SP)      ;PUSH DOWN THE PC
 7395  026202  016666  000004  000002 MOV     4(SP),2(SP)     ;SAVE THE PS
 7396  026210  105777  152722 1$:     TSTB    @$TKS           ;WAIT FOR
 7397  026214  100375                 BPL     1$              ;A CHARACTER
 7398  026216  117766  152716  000004 MOVB    @$TKB,4(SP)     ;READ THE TTY
 7399  026224  042766  177600  000004 BIC     #^C<177>,4(SP)  ;GET RID OF JUNK IF ANY
 7400  026232  000002                 RTI                     ;GO BACK TO USER
 7401
 7402                         ;*****************************************************************
 7403                         ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 7404                         ;*CALL:
 7405                         ;*      RDLIN                   ;INPUT A STRING FROM THE TTY
 7406                         ;*      RETURN HERE             ;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 7407                         ;*                              ;TERMINATOR WILL BE A BYTE OF ALL 0'S
 7408
 7409  026234  010346         $RDLIN: MOV     R3,-(SP)        ;SAVE R3
 7410  026236  005046                 CLR     -(SP)           ;CLEAR THE RUBOUT KEY
 7411  026240  012703  026502 1$:     MOV     #$TTYIN,R3      ;GET ADDRESS
 7412  026244  022703  026514 2$:     CMP     #$TTYIN+10.,R3  ;BUFFER FULL?
 7413  026250  101450                 BLOS    4$              ;BR IF YES
 7414  026252  104412                 RDCHR                   ;GO READ ONE CHARACTER FROM THE TTY
 7415  026254  112613                 MOVB    (SP)+,(R3)      ;GET CHARACTER
 7416  026256  122713  000177         CMPB    #177,(R3)       ;IS IT A RUBOUT
 7417  026262  001022                 BNE     5$              ;BR IF NO
 7418  026264  005716                 TST     (SP)            ;IS THIS THE FIRST RUBOUT?
 7419  026266  001007                 BNE     6$              ;BR IF NO
 7420  026270  112737  000134  026466 MOVB    #'\,9$          ;TYPE A BACK SLASH
 7421  026276  104400  026466         TYPE    ,9$
 7422  026302  012716  177777         MOV     #-1,(SP)        ;SET THE RUBOUT KEY
 7423  026306  005303         6$:     DEC     R3              ;BACKUP BY ONE
 7424  026310  020327  026502         CMP     R3,#$TTYIN      ;STACK EMPTY?
 7425  026314  103426                 BLO     4$              ;BR IF YES
 7426  026316  111337  026466         MOVB    (R3),9$         ;SETUP TO TYPEOUT THE DELETED CHAR.
 7427  026322  104400  026466         TYPE    ,9$             ;GO TYPE
 7428  026326  000746                 BR      2$              ;GO READ ANOTHER CHAR.
 7429  026330  005716         5$:     TST     (SP)            ;RUBOUT KEY SET?
 7430  026332  001406                 BEQ     7$              ;BR IF NO
 7431  026334  112737  000134  026466 MOVB    #'\,9$          ;TYPE A BACK SLASH
 7432  026342  104400  026466         TYPE    ,9$
 7433  026346  005016                 CLR     (SP)            ;CLEAR THE RUBOUT KEY
```

```
 7434  026350  122713  000003          7$:     CMPB    #3,(R3) ;IS CHARACTER A CTRL C ?
 7435  026354  001425                          BEQ     8$              ;BR IF IT IS
 7436  026356  122713  000025                  CMPB    #25,(R3)        ;IS CHARACTER A CTRL U?
 7437  026362  001006                          BNE     3$              ;BR IF NO
 7438  026364  104400  026470                  TYPE    ,$CNTLU         ;TYPE A CONTROL "U"
 7439  026370  000723                          BR      1$              ;GO START OVER
 7440  026372  104400  001156          4$:     TYPE    ,$QUES          ;TYPE A '?'
 7441  026376  000720                          BR      1$              ;CLEAR THE BUFFER AND LOOP
 7442  026400  111337  026466          3$:     MOVB    (R3),9$         ;ECHO THE CHARACTER
 7443  026404  104400  026466                  TYPE    ,9$
 7444  026410  122723  000015                  CMPB    #15,(R3)+       ;CHECK FOR RETURN
 7445  026414  001313                          BNE     2$              ;LOOP IF NOT RETURN
 7446  026416  105063  177777                  CLRB    -1(R3)          ;CLEAR RETURN (THE 15)
 7447  026422  104400  001160                  TYPE    ,$LF            ;TYPE A LINE FEED
 7448  026426  000405                          BR      10$
 7449  026430  104400  026475          8$:     TYPE    ,$CNTLC         ;TYPE A CONTROL C
 7450  026434  112737  000003  026502          MOVB    #3,$TTYIN       ;FORCE 1ST CHARACTER IN BUF TO CTRL C
 7451  026442  005726                  10$:    TST     (SP)+           ;CLEAN RUBOUT KEY FROM THE STACK
 7452  026444  012603                          MOV     (SP)+,R3        ;RESTORE R3
 7453  026446  011646                          MOV     (SP),-(SP)      ;ADJUST THE STACK AND PUT ADDRESS OF THE
 7454  026450  016666  000004  000002          MOV     4(SP),2(SP)     ;       FIRST ASCII CHARACTER ON IT
 7455  026456  012766  026502  000004          MOV     #$TTYIN,4(SP)
 7456  026464  000002                          RTI                     ;RETURN
 7457  026466     000                  9$:     .BYTE   0               ;STORAGE FOR ASCII CHAR. TO TYPE
 7458  026467     000                          .BYTE   0               ;TERMINATOR
 7459  026470  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>    ;CONTROL "U"
 7460  026475     136  006503  000012  $CNTLC: .ASCIZ  /↑C/<15><12>    ;CONTROL C
 7461  026502  000012                  $TTYIN: .BLKB   10.             ;RESERVE 10. BYTES FOR TTY INPUT
 7462                                          .EVEN
 7463
 7464                                  ;******************************************************************
 7465
 7466                                  .SBTTL  MACRO ROUTINES
 7467
 7468                                  ;******************************************************************
  (1)
  (1)                                 .SBTTL  ERROR HANDLER ROUTINE
  (1)
  (1)                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
  (1)                                 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
  (1)                                 ;*AND GO TO $ERRTYP ON ERROR
  (1)                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                 ;*SW15=1        HALT ON ERROR
  (1)                                 ;*SW13=1        INHIBIT ERROR TYPEOUTS
  (1)                                 ;*SW10=1        BELL ON ERROR
  (1)                                 ;*CALL
  (1)                                 ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
  (1)
  (1)  026514                         $ERROR:
  (2)  026514  004737  025170                 7$:     JSR     PC,ERENTR
  (1)  026520  105237  001103                 7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
  (1)  026524  001775                         BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
  (1)  026526  013737  001102  177570         MOV     $TSTNM,@#DISPLAY;;DISPLAY TEST NUMBER AND ERROR FLAG
  (1)  026534  032737  002000  177570         BIT     #BIT10,@#SWR    ;;BELL ON ERROR?
```

```
(1)  026542  001402                        BEQ     1$                ;;NO - SKIP
(1)  026544  104400  001152                TYPE    ,$BELL            ;;RING BELL
(1)  026550  005237  001112          1$:   INC     $ERTTL            ;;COUNT THE NUMBER OF ERRORS
(1)  026554  011637  001116                MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  026560  162737  000002  001116        SUB     #2,$ERRPC
(1)  026566  117737  152324  001114        MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  026574  032737  020000  177570        BIT     #BIT13,@#SWR      ;;SKIP TYPEOUT IF SET
(1)  026602  001004                        BNE     2$                ;;SKIP TYPEOUTS
(1)  026604  004737  026626                JSR     PC,@#$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  026610  104400  001157                TYPE    ,$CRLF
(1)  026614  005737  177570          2$:   TST     @#SWR             ;;HALT ON ERROR
(1)  026620  100001                        BPL     3$                ;;SKIP IF CONTINUE
(1)  026622  000000                        HALT                      ;;HALT ON ERROR!
(1)  026624                          3$:
(1)  026624  000002                        RTI                       ;;RETURN
7469                                ;*****************************************************************
 (1)
 (1)                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 (1)
 (1)                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 (1)                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 (1)                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 (1)
(1)  026626                         $ERRTYP:
(1)  026626  104400  001157                TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  026632  010046                        MOV     R0,-(SP)          ;;SAVE R0
(1)  026634  005000                        CLR     R0                ;;PICKUP THE ITEM INDEX
(1)  026636  153700  001114                BISB    @#$ITEMB,R0
(1)  026642  001004                        BNE     1$                ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  026644  013746  001116                MOV     $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  026650  104402                        TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  026652  000445                        BR      10$               ;;GET OUT
(1)  026654  005300                  1$:   DEC     R0                ;;ADJUST THE INDEX SO THAT IT WILL
(1)  026656  006300                        ASL     R0                ;;      WORK FOR THE ERROR TABLE
(1)  026660  006300                        ASL     R0
(1)  026662  006300                        ASL     R0
(1)  026664  062700  001340                ADD     #$ERRTB,R0        ;;FORM TABLE POINTER
(1)  026670  012037  026700                MOV     (R0)+,2$          ;;PICKUP "ERROR MESSAGE" POINTER
(1)  026674  001404                        BEQ     3$                ;;SKIP TYPEOUT IF NO POINTER
(1)  026676  104400                        TYPE                      ;;TYPE THE "ERROR MESSAGE"
(1)  026700  000000                  2$:   .WORD   0                 ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  026702  104400  001157                TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  026706  012037  026716          3$:   MOV     (R0)+,4$          ;;PICKUP "DATA HEADER" POINTER
(1)  026712  001404                        BEQ     5$                ;;SKIP TYPEOUT IF 0
(1)  026714  104400                        TYPE                      ;;TYPE THE "DATA HEADER"
(1)  026716  000000                  4$:   .WORD   0                 ;;"DATA HEADER" POINTER GOES HERE
(1)  026720  104400  001157                TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  026724  010146                  5$:   MOV     R1,-(SP)          ;;SAVE R1
(1)  026726  012001                        MOV     (R0)+,R1          ;;PICKUP "DATA TABLE" POINTER
(1)  026730  001415                        BEQ     9$                ;;BR IF NO DATA TO BE TYPED
(1)  026732  012000                        MOV     (R0)+,R0          ;;PICKUP "DATA FORMAT" POINTER
(1)  026734  105720                  6$:   TSTB    (R0)+             ;;"OCTAL" OR "DECIMAL"
```

```
(1)  026736  001003                        BNE     7$                ;;BR IF DECIMAL
(2)  026740  013146                        MOV     @(R1)+,-(SP)      ;;SAVE @(R1)+ FOR TYPEOUT
(2)  026742  104402                        TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  026744  000402                        BR      8$
(2)  026746                          7$:
(2)  026746  013146                        MOV     @(R1)+,-(SP)      ;;SAVE @(R1)+ FOR TYPEOUT
(2)  026750  104410                        TYPOS                     ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  026752  005711                  8$:   TST     (R1)              ;;IS THERE ANOTHER NUMBER?
(1)  026754  001403                        BEQ     9$                ;;BR IF NO
(1)  026756  104400  026776                TYPE    ,11$              ;;TYPE TWO(2) SPACES
(1)  026762  000764                        BR      6$                ;;LOOP
(1)
(1)  026764  012601                  9$:   MOV     (SP)+,R1          ;;RESTORE R1
(1)  026766  012600                  10$:  MOV     (SP)+,R0          ;;RESTORE R0
(1)  026770  104400  001157                TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  026774  000207                        RTS     PC                ;;RETURN
(1)  026776  020040      000         11$:  .ASCIZ  / /               ;;TWO(2) SPACES
(1)          027002                        .EVEN
7470                                ;*****************************************************************
(1)
(1)                                 .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)         ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPOS                     ;;CALL FOR TYPEOUT
(1)                                 ;*      .BYTE   N                 ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                 ;*      .BYTE   M                 ;;M=1 OR 0
(1)                                 ;*                                ;;1=TYPE LEADING ZEROS
(1)                                 ;*                                ;;0=SUPPRESS LEADING ZEROS
(1)                                 ;*
(1)                                 ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                 ;*$TYPOS OR $TYPOC
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)         ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPON                     ;;CALL FOR TYPEOUT
(1)                                 ;*
(1)                                 ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)         ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPOC                     ;;CALL FOR TYPEOUT
(1)
(1)  027002  017646  000000        $TYPOS: MOV     @(SP),-(SP)       ;;PICKUP THE MODE
(1)  027006  116637  000001  027225        MOVB    1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
(1)  027014  112637  027227                MOVB    (SP)+,$OMODE+1    ;;NUMBER OF DIGITS TO TYPE
(1)  027020  062716  000002                ADD     #2,(SP)           ;;ADJUST RETURN ADDRESS
(1)  027024  000406                        BR      $TYPON
(1)  027026  112737  000001  027225  $TYPOC: MOVB    #1,$OFILL         ;;SET THE ZERO FILL SWITCH
(1)  027034  112737  000006  027227        MOVB    #6,$OMODE+1       ;;SET FOR SIX(6) DIGITS
(1)  027042  112737  000005  027224  $TYPON: MOVB    #5,$OCNT          ;;SET THE ITERATION COUNT
(1)  027050  010346                        MOV     R3,-(SP)          ;;SAVE R3
(1)  027052  010446                        MOV     R4,-(SP)          ;;SAVE R4
```

```
(1)  027054  010546                        MOV     R5,-(SP)        ;;SAVE R5
(1)  027056  113704  027227                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  027062  005404                        NEG     R4
(1)  027064  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  027070  110437  027226                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  027074  113704  027225                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  027100  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  027104  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  027106  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  027110  000404                        BR      3$              ;;GO DO MSB
(1)  027112  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  027114  006105                        ROL     R5
(1)  027116  006105                        ROL     R5
(1)  027120  010503                        MOV     R5,R3
(1)  027122  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  027124  105337  027226                DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  027130  100016                        BPL     7$              ;;BR IF NO
(1)  027132  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
(1)  027136  001002                        BNE     4$              ;;TEST FOR 0
(1)  027140  005704                        TST     R4              ;;SUPPRESS THIS 0?
(1)  027142  001403                        BEQ     5$              ;;BR IF YES
(1)  027144  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  027146  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  027152  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  027156  110337  027222                MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  027162  104400  027222                TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  027166  105337  027224        7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  027172  003347                        BGT     2$              ;;BR IF MORE TO DO
(1)  027174  002402                        BLT     6$              ;;BR IF DONE
(1)  027176  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  027200  000744                        BR      2$              ;;GO DO THE LAST DIGIT
(1)  027202  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  027204  012604                        MOV     (SP)+,R4        ;;RESTORE R4
(1)  027206  012603                        MOV     (SP)+,R3        ;;RESTORE R3
(1)  027210  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  027216  012616                        MOV     (SP)+,(SP)
(1)  027220  000002                        RTI                     ;;RETURN
(1)  027222     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  027223     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  027224     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  027225     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  027226  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
7471                                ;******************************************************************
(1)
(1)                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                                 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                                 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                                 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                                 ;*REPLACED WITH SPACES.
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                                 ;*      TYPOS                   ;;GO TO THE ROUTINE
```

```
(1)
(2)  027230                          $TYPDS:
(3)  027230  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  027232  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  027234  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  027236  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  027240  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  027242  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  027246  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  027252  100004                          BPL     1$              ;;BR IF INPUT IS POS.
(1)  027254  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  027256  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  027264  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  027266  012703  027444                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  027272  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  027276  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  027300  016001  027434                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  027304  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  027306  002402                          BLT     4$              ;;BR IF DONE
(1)  027310  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  027312  000774                          BR      3$
(1)  027314  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  027316  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  027320  001002                          BNE     5$              ;;FALL THROUGH IF 0
(1)  027322  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  027324  100407                          BMI     7$              ;;BR IF YES
(1)  027326  106316                  5$:     ASLB    (SP)            ;;MSD?
(1)  027330  103003                          BCC     6$              ;;BR IF NO
(1)  027332  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  027340  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  027344  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  027350  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  027352  005720                          TST     (R0)+           ;;JUST INCREMENTING
(1)  027354  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  027360  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  027362  003002                          BGT     8$              ;;GO TO EXIT
(1)  027364  010502                          MOV     R5,R2           ;;GET THE LSD
(1)  027366  000764                          BR      6$              ;;GO CHANGE TO ASCII
(1)  027370  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  027372  100003                          BPL     9$              ;;BR IF NO
(1)  027374  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  027402  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  027404  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  027406  012503                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  027410  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  027412  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  027414  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  027416  104400  027444                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  027422  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  027430  012616                          MOV     (SP)+,(SP)
(1)  027432  000002                          RTI                     ;;RETURN TO USER
(1)  027434  023420                  $DTBL:  10000.
(1)  027436  001750                          1000.
(1)  027440  000144                          100.
```

```
  (1)  027442  000012                        10.
  (1)  027444  000004                $DBLK:  .BLKW   4
 7472                                ;*****************************************************************
  (1)
  (1)                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
  (1)
  (1)                                ;*SAVE R0-R5
  (1)                                ;*CALL:
  (1)                                ;*      SAVREG
  (1)                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
  (1)                                ;*
  (1)                                ;*TOP---(+16)
  (1)                                ;* +2---(+18)
  (1)                                ;* +4---R5
  (1)                                ;* +6---R4
  (1)                                ;* +8---R3
  (1)                                ;*+10---R2
  (1)                                ;*+12---R1
  (1)                                ;*+14---R0
  (1)
  (1)  027454                        $SAVREG:
  (3)  027454  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  027456  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  027460  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  027462  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  027464  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)  027466  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (1)  027470  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
  (1)  027474  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
  (1)  027500  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
  (1)  027504  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
  (1)  027510  000002                        RTI
  (1)
  (1)                                ;*RESTORE R0-R5
  (1)                                ;*CALL:
  (1)                                ;*      RESREG
  (1)  027512                        $RESREG:
  (1)  027512  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
  (1)  027516  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
  (1)  027522  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
  (1)  027526  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
  (3)  027532  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  027534  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
  (3)  027536  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  027540  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  027542  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  027544  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  027546  000002                        RTI
 7473                                ;*****************************************************************
  (1)
  (1)                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
  (1)
  (1)                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
  (1)                                ;*WITH A RANGE OF 0 TO 2(+33)-1.
```

```
(1)                                     ;*CALL:
(1)                                     ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
(1)                                     ;*      RETURN                  ;;RETURN HERE THE RANDOM
(1)                                     ;*                              ;;NUMBER WILL BE IN
(1)                                     ;*                              ;;$HINUM,$LONUM
(1)
(2)  027550                             $RAND:
(3)  027550  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  027552  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  027554  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  027556  010346                             MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(1)  027560  013700  027676                     MOV     $LONUM,R0       ;;SET R0 WITH LOW
(1)  027564  013701  027674                     MOV     $HINUM,R1       ;;SET R1 WITH HIGH
(1)  027570  012703  177771                     MOV     #-7,R3          ;;SET SHIFT COUNT
(1)  027574  005002                             CLR     R2              ;;ZERO R2
(1)  027576  006300                     1$:     ASL     R0              ;;SHIFT R0 LEFT AND
(1)  027600  006101                             ROL     R1              ;;ROTATE CARRY INTO R1 AND
(1)  027602  006102                             ROL     R2              ;;ROTATE CARRY INTO R2
(1)  027604  005203                             INC     R3              ;;CHECK FOR DONE
(1)  027606  001373                             BNE     1$              ;;CONTINUE SHIFT LOOP
(1)  027610  063700  027676                     ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
(1)  027614  005501                             ADC     R1              ;;PROPOGATE CARRY
(1)  027616  063701  027674                     ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
(1)  027622  005502                             ADC     R2              ;;PROPOGATE CARRY
(1)  027624  062700  001057                     ADD     #1057,R0        ;;ADD LOW CONSTANT
(1)  027630  005501                             ADC     R1              ;;PROPOGATE CARRY
(1)  027632  005502                             ADC     R2              ;;PROPOGATE CARRY
(1)  027634  062701  047401                     ADD     #47401,R1       ;;ADD HIGH CONSTANT
(1)  027640  005502                             ADC     R2              ;;PROPOGATE CARRY
(1)  027642  062702  000006                     ADD     #6,R2           ;;ADD HIGHEST CONSTART
(1)  027646  060200                             ADD     R2,R0           ;;REPRIME R0 WITH HIGHEST DIGIT
(1)  027650  005501                             ADC     R1              ;;PROPOGATE CARRY
(1)  027652  010037  027676                     MOV     R0,$LONUM       ;;SAVE R0
(1)  027656  010137  027674                     MOV     R1,$HINUM       ;;SAVE R1
(3)  027662  012603                             MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  027664  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  027666  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  027670  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  027672  000207                             RTS     PC              ;;RETURN
(1)  027674  176543                     $HINUM: .WORD   176543
(1)  027676  123456                     $LONUM: .WORD   123456
7474                                    ;******************************************************************
(1)
(1)                                     .SBTTL  ROUTINE TO SIZE MEMORY
(1)
(1)                                     ;*CALL:
(1)                                     ;*      JSR     PC,$SIZE
(1)                                     ;*      RETURN
(1)                                     ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
(1)
(1)  027700  010046                     $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
(1)  027702  010146                             MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
(1)  027704  013746  000004                     MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
(1)  027710  013746  000006                     MOV     @#ERRVEC+2,-(SP)
```

```
   (1)  027714  010600                        MOV     SP,R0           ;;SAVE THE STACK POINTER
   (1)  027716  013737  177776  000006        MOV     @#PS,@#ERRVEC+2 ;;SET ERRVEC PS TO PRESENT PS
   (1)  027724  012737  027742  000004        MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
   (1)  027732  012701  020000                MOV     #20000,R1       ;;FIRST ADDRESS
   (1)  027736  005721                1$:     TST     (R1)+           ;;TEST THIS ADDRESS
   (1)  027740  000776                        BR      1$              ;;TRY ANOTHER
   (1)  027742  162701  000004        2$:     SUB     #4,R1           ;;DROP BACK
   (1)  027746  010006                        MOV     R0,SP           ;;RESTORE THE STACK
   (1)  027750  012637  000006                MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
   (1)  027754  012637  000004                MOV     (SP)+,@#ERRVEC
   (1)  027760  010137  027772                MOV     R1,$LSTAD       ;;LAST ADDRESS
   (1)  027764  012601                        MOV     (SP)+,R1        ;;RESTORE R1
   (1)  027766  012600                        MOV     (SP)+,R0        ;;RESTORE R0
   (1)  027770  000207                        RTS     PC
   (1)  027772  000000                $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
 7475                                 ;*****************************************************************
   (1)                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
   (1)
   (1)                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
   (1)                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
   (1)                                ;*POSITIVE.
   (1)                                ;*CALL
   (1)                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
   (1)                                ;*      JSR     PC,@#$DB2D
   (1)                                ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
   (1)                                                                ;;IS ON THE STACK
   (1)
   (1)
   (1)  027774  104416                $DB2D:  SAVREG                  ;;SAVE REGISTERS
   (1)  027776  016602  000002                MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
   (1)  030002  012700  030154                MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
   (1)  030006  010066  000002                MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
   (1)  030012  012201                        MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
   (1)  030014  012202                        MOV     (R2)+,R2
   (1)  030016  012737  000012  030072        MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
   (1)  030024  012704  030104                MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
   (1)  030030  012705  030106                MOV     #$TNPWR+2,R5
   (1)  030034  005003                1$:     CLR     R3              ;;CLEAR PARTIAL
   (1)  030036  161401                2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
   (1)  030040  005602                        SBC     R2
   (1)  030042  161502                        SUB     (R5),R2
   (1)  030044  002402                        BLT     3$              ;;BR IF TEN POWER TO LARGE
   (1)  030046  005203                        INC     R3              ;;ADD 1 TO PARTIAL
   (1)  030050  000772                        BR      2$              ;;LOOP
   (1)  030052  062401                3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
   (1)  030054  005502                        ADC     R2
   (1)  030056  062402                        ADD     (R4)+,R2
   (1)  030060  022525                        CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
   (1)  030062  052703  000060                BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
   (1)  030066  110320                        MOVB    R3,(R0)+        ;;SAVE IT
   (1)  030070  005327                        DEC     (PC)+           ;;DONE?
   (1)  030072  000000                4$:     .WORD   0
   (1)  030074  001357                        BNE     1$              ;;BR IF NO
```

```
(1)  030076  105020                          CLRB    (R0)+           ;;TERMINATOR
(1)  030100  104420                          RESREG                  ;;RESTORE REGISTERS
(1)  030102  000207                          RTS     PC              ;;RETURN
(1)  030104  145000                $TNPWR:   145000                  ;;1.0E09
(1)  030106  035632                          35632
(1)  030110  160400                          160400                  ;;1.0E08
(1)  030112  002765                          2765
(1)  030114  113200                          113200                  ;;1.0E07
(1)  030116  000230                          230
(1)  030120  041100                          041100                  ;;1.0E06
(1)  030122  000017                          17
(1)  030124  103240                          103240                  ;;1.0E05
(1)  030126  000001                          1
(1)  030130  023420                          23420                   ;;1.0E04
(1)  030132  000000                          0
(1)  030134  001750                          1750                    ;;1.0E03
(1)  030136  000000                          0
(1)  030140  000144                          144                     ;;1.0E02
(1)  030142  000000                          0
(1)  030144  000012                          12                      ;;1.0E01
(1)  030146  000000                          0
(1)  030150  000001                          1                       ;;1.0E00
(1)  030152  000000                          0
(1)  030154  000014                $DECVL:   .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
7476                                         ;*********************************************************************
(1)
(1)                                          .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
(1)
(1)                                          ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
(1)                                          ;*UNSIGNED DECIMAL ASCIZ NUMBER.
(1)                                          ;*CALL
(1)                                          ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
(1)                                          ;*      JSR     PC,@#$SB2D      ;;CALL
(1)                                          ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
(1)
(1)
(1)  030170  016637  000002  030214  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
(1)  030176  012746  030214                  MOV     #1$,-(SP)       ;;SET POINTER
(1)  030202  004737  027774                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
(1)  030206  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
(1)  030212  000207                          RTS     PC              ;;RETURN
(1)  030214  000000  000000          1$:     .WORD   0,0
7477                                         ;*********************************************************************
(1)
(1)                                          .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
(1)
(1)                                          ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
(1)                                          ;*UNSIGNED OCTAL ASCIZ NUMBER.
(1)                                          ;*CALL
(1)                                          ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
(1)                                          ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
(1)                                          ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
(1)
(1)
```

```
(1)  030220  104416                $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
(1)  030222  016601  000002                MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
(1)  030226  012705  030337                MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
(1)  030232  012704  000014                MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
(1)  030236  012703  177770                MOV     #177770,R3      ;;MASK
(1)  030242  012100                        MOV     (R1)+,R0        ;;LOWER WORD
(1)  030244  012101                        MOV     (R1)+,R1        ;;HIGH WORD
(1)  030246  005002                        CLR     R2              ;;TERMINATOR
(1)  030250  110245                1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
(1)  030252  010002                        MOV     R0,R2           ;;GET THIS DIGIT
(1)  030254  005304                        DEC     R4              ;;COUNT THIS CHARACTER
(1)  030256  003007                        BGT     3$              ;;BR IF NOT THE LAST DIGIT
(1)  030260  001405                        BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
(1)  030262  005205                        INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
(1)  030264  010566  000002                MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
(1)  030270  104420                        RESREG                  ;;RESTORE ALL REGISTERS
(1)  030272  000207                        RTS     PC              ;;RETURN TO USER
(1)  030274  006203                2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
(1)  030276  006001                3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
(1)  030300  006000                        ROR     R0              ;;      THE NEXT OCTAL DIGIT
(1)  030302  006001                        ROR     R1
(1)  030304  006000                        ROR     R0
(1)  030306  006001                        ROR     R1
(1)  030310  006000                        ROR     R0
(1)  030312  040302                        BIC     R3,R2           ;;MASK OUT ALL JUNK
(1)  030314  062702  000060                ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
(1)  030320  000753                        BR      1$              ;;GO PUT IT IN THE DATA TABLE
(1)  030322  000016                $OCTVL: .BLKB   14.              ;;RESERVE DATA TABLE
7478                               ;*****************************************************************
(1)
(1)                                .SBTTL  SINGLE LENGTH BINARY TO OCTAL ASCIZ ROUTINE
(1)
(1)                                ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
(1)                                ;*UNSIGNED OCTAL ASCIZ NUMBER.
(1)                                ;*CALL
(1)                                ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
(1)                                ;*      JSR     PC,@#$SB20      ;;CALL
(1)                                ;*      RETURN                  ;;ADDRESS OF 1ST ASCIZ CHAR. IS ON THE STACK
(1)
(1)  030340  016637  000002  030364 $SB20:  MOV     2(SP),1$        ;;SAVE THE BINARY NUMBER
(1)  030346  012746  030364                MOV     #1$,-(SP)       ;;SET POINTER
(1)  030352  004737  030220                JSR     PC,@#$DB20      ;;CALL DOUBLE LENGTH CONVERT ROUTINE
(1)  030356  012666  000002                MOV     (SP)+,2(SP)     ;;PICKUP POINTER
(1)  030362  000207                        RTS     PC              ;;RETURN
(1)  030364  000000  000000        1$:     .WORD   0,0
7479                               ;*****************************************************************
(1)
(1)                                .SBTTL  TRAP DECODER
(1)
(1)                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                                ;*GO TO THAT ROUTINE.
(1)
```

```
(1)  030370  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  030372  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  030376  005740                          TST     -(R0)           ;;BACKUP BY 2
(1)  030400  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  030402  022700  000024                  CMP     #$TERM,R0       ;;CHECK FOR OUT OF BOUNDS
(1)  030406  003002                          BGT     .+6             ;;BR IF OK
(1)  030410  000000                          HALT                    ;;OUT OF BOUNDS
(1)  030412  000776                          BR      .-2             ;;HANGUP
(1)  030414  016000  030422                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  030420  000200                          RTS     R0              ;;GO TO ROUTINE
(1)
(3)
(3)                                  .SBTTL  TRAP TABLE
(3)
(3)                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                                  ;*BY THE "TRAP" INSTRUCTION.
(3)
(3)                                  ;       ROUTINE
(3)                                  ;       -------
(3)  030422                          $TRPAD:
(3)  030422  025550                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
(3)  030424  027026                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  030426  027002                          $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  030430  027042                          $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  030432  027230                          $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
(3)  030434  026200                          $RDCHR  ;;CALL=RDCHR    TRAP+12(104412) TTY TYPEIN CHARACTER ROUTINE
(3)  030436  026234                          $RDLIN  ;;CALL=RDLIN    TRAP+14(104414) TTY TYPEIN STRING ROUTINE
(3)  030440  027454                          $SAVREG ;;CALL=SAVREG   TRAP+16(104416) SAVE R0-R5 ROUTINE
(3)  030442  027512                          $RESREG ;;CALL=RESREG   TRAP+20(104420) RESTORE R0-R5 ROUTINE
7480 030444  025074                          $DSPLY  ;;CALL=DISPLY   TRAP+22(104422) PRINTER-TELETYPE MESSAGE ROUTING ROUTINE
7481         000024                  $TERM=.-$TRPAD
7482
7483                                 ;*****************************************************************
7484
7485                                 .SBTTL  RH11/RP04 DRIVER - SINGLE/DUAL PORT VERSION
7486                                         ;(MODIFIED FROM REV. 0.3 FOR MD-11-DZRPN-B)
7487
7488                                 ;*COPYRIGHT (C) 1974
7489                                 ;*DIGITAL EQUIPMENT CORP.
7490                                 ;*MAYNARD, MA 01754
7491                                 ;*AUTHOR: JIM LACEY/CHUCK HESS
7492
7493                                 ;*****************************************************************
7494
7495                                 ;*STORAGE FOR RHDS1, RHER1, RHER2, AND RHER3 ON AN ERROR "2" OR "5"
7496                                         ;*RPERRS   = RHDS1
7497                                         ;*RPERRS+2 = RHER1
7498                                         ;*RPERRS+4 = RHER2
7499                                         ;*RPERRS+6 = RHER3
7500
7501 030446  000000                  RPERRS: .WORD   0
7502 030450  000000                          .WORD   0
7503 030452  000000                          .WORD   0
7504 030454  000000                          .WORD   0
```

```
7505
7506                                    ;*TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)
7507                                    ;*DRVACT=0 IMPLIES DRIVE IS IDLE
7508                                    ;*DRVACT>0 IMPLIES DRIVE IS ACTIVE WITH A COMMAND
7509                                    ;*DRVACT<0 IMPLIES DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION
7510
7511  030456  000                   DRVACT: .BYTE   0               ;DRIVE 0
7512  030457  000                           .BYTE   0               ;DRIVE 1
7513  030460  000                           .BYTE   0               ;DRIVE 2
7514  030461  000                           .BYTE   0               ;DRIVE 3
7515  030462  000                           .BYTE   0               ;DRIVE 4
7516  030463  000                           .BYTE   0               ;DRIVE 5
7517  030464  000                           .BYTE   0               ;DRIVE 6
7518  030465  000                           .BYTE   0               ;DRIVE 7
7519
7520                                    ;*TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
7521                                    ;*DRVSTA=0 IMPLIES DRIVE IS OFFLINE
7522                                    ;*DRVSTA>0 IMPLIES DRIVE IS ONLINE
7523                                    ;*DRVSTA<0 IMPLIES DRIVE IS UNSAFE OR NONEXISTENT
7524
7525  030465  000                   DRVSTA: .BYTE   0               ;DRIVE 0
7526  030467  000                           .BYTE   0               ;DRIVE 1
7527  030470  000                           .BYTE   0               ;DRIVE 2
7528  030471  000                           .BYTE   0               ;DRIVE 3
7529  030472  000                           .BYTE   0               ;DRIVE 4
7530  030473  000                           .BYTE   0               ;DRIVE 5
7531  030474  000                           .BYTE   0               ;DRIVE 6
7532  030475  000                           .BYTE   0               ;DRIVE 7
7533
7534                                    ;*TABLE OF DRIVE TYPES (DRVTYP=8 WORDS)
7535                                    ;*DRVTYP WILL CONTAIN THE DRIVE TYPE OF ALL ONLINE, OFFLINE, AND
7536                                    ;*UNSAFE DRIVES. IF A DRIVE IS NONEXISTENT DRVTYP WILL BE ZERO.
7537
7538  030476  000000                DRVTYP: .WORD   0               ;DRIVE 0
7539  030500  000000                        .WORD   0               ;DRIVE 1
7540  030502  000000                        .WORD   0               ;DRIVE 2
7541  030504  000000                        .WORD   0               ;DRIVE 3
7542  030506  000000                        .WORD   0               ;DRIVE 4
7543  030510  000000                        .WORD   0               ;DRIVE 5
7544  030512  000000                        .WORD   0               ;DRIVE 6
7545  030514  000000                        .WORD   0               ;DRIVE 7
7546
7547                                    ;*TABLE OF DUAL PORT INITIALIZATION INDICATORS
7548                                    ;*DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
7549                                    ;*DPINT<0 IF INITIALIZATION IS IN PROGRESS
7550
7551  030516  000                   DPINT:  .BYTE   0               ;DRIVE 0
7552  030517  000                           .BYTE   0               ;DRIVE 1
7553  030520  000                           .BYTE   0               ;DRIVE 2
7554  030521  000                           .BYTE   0               ;DRIVE 3
7555  030522  000                           .BYTE   0               ;DRIVE 4
7556  030523  000                           .BYTE   0               ;DRIVE 5
7557  030524  000                           .BYTE   0               ;DRIVE 6
7558  030525  000                           .BYTE   0               ;DRIVE 7
```

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

030526  000             DPRQS:  .BYTE   0               ;DRIVE 0
030527  000                     .BYTE   0               ;DRIVE 1
030530  000                     .BYTE   0               ;DRIVE 2
030531  000                     .BYTE   0               ;DRIVE 3
030532  000                     .BYTE   0               ;DRIVE 4
030533  000                     .BYTE   0               ;DRIVE 5
030534  000                     .BYTE   0               ;DRIVE 6
030535  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.

030536  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 PERFOR ING 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.

030540  000000          SRCHWT: .WORD   0

                                ;*RP04 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
                                        ;*ACTDRV=0 IMPLIES DRIVER IS INACTIVE
                                        ;*ACTDRV>0 IMPLIES DRIVER IS ACTIVE

030542  000             ACTDRV: .BYTE   0

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

030543  000             ACTSTR: .BYTE   0

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

030544  000             ULDFLG: .BYTE   0               ;DRIVE 0
030545  000                     .BYTE   0               ;DRIVE 1
030546  000                     .BYTE   0               ;DRIVE 2
030547  000                     .BYTE   0               ;DRIVE 3
030550  000                     .BYTE   0               ;DRIVE 4
030551  000                     .BYTE   0               ;DRIVE 5
030552  000                     .BYTE   0               ;DRIVE 6
030553  000                     .BYTE   0               ;DRIVE 7
```

```
7613                                    ;*LOOK AHEAD COUNT (LACNT=8 BYTES)
7614                                            ;*LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
7615
7616  030554    000                     LACNT:  .BYTE   0               ;DRIVE 0
7617  030555    000                             .BYTE   0               ;DRIVE 1
7618  030556    000                             .BYTE   0               ;DRIVE 2
7619  030557    000                             .BYTE   0               ;DRIVE 3
7620  030560    000                             .BYTE   0               ;DRIVE 4
7621  030561    000                             .BYTE   0               ;DRIVE 5
7622  030562    000                             .BYTE   0               ;DRIVE 6
7623  030563    000                             .BYTE   0               ;DRIVE 7
7624
7625                                    ;*SAVE REGISTERS FLAG (SAVEFG =1 WORD)
7626                                            ;*SAVEFG <0 IMPLIES SAVE THE RH11/RP04 REGISTERS WHEN THE
7627                                            ;*OPERATION IS COMPLETED AS PER (DPB+14).
7628                                            ;*SAVEFG=0 IMPLIES SAVE THE RH11/RP04 REGISTERS, AS PER
7629                                            ;*(DPB+14), AFTER AN ERROR.
7630
7631  030564  000000                    SAVEFG: .WORD   0
7632
7633                                    ;*SEEK FLAG (SEEKFG=1 WORD)
7634                                            ;*SEEKFG=0 IMPLIES WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
7635                                            ;*FOR A DATA TRANSFER START A SEARCH COMMAND
7636                                            ;*SEEKFG<0 IMPLIES DATA TRANSFER WILL DO IMPLIED SEEKS,
7637                                            ;*DISREGARD THE WINDOW
7638
7639  030566  000000                    SEEKFG: .WORD   0
7640
7641                                    ;*TIMEOUT TABLE (TIMER=8 WORDS)
7642                                            ;*THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
7643
7644  030570  177777                    TIMER:  .WORD   -1              ;DRIVE 0
7645  030572  177777                            .WORD   -1              ;DRIVE 1
7646  030574  177777                            .WORD   -1              ;DRIVE 2
7647  030576  177777                            .WORD   -1              ;DRIVE 3
7648  030600  177777                            .WORD   -1              ;DRIVE 4
7649  030602  177777                            .WORD   -1              ;DRIVE 5
7650  030604  177777                            .WORD   -1              ;DRIVE 6
7651  030606  177777                            .WORD   -1              ;DRIVE 7
7652
7653                                    ;*DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
7654                                            ;*DTUW<0 IMPLIES NO DATA TRANSFER UNDERWAY
7655                                            ;*DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
7656
7657  030610  177777                    DTUW:   .WORD   -1
7658
7659                                    ;*ATTENTION BITS TABLE (ATABIT=8 BYTES)
7660                                            ;*THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
7661                                            ;*ATTENTION BIT
7662
7663  030612    001                     ATABIT: .BYTE   1               ;DRIVE 0
7664  030613    002                             .BYTE   2               ;DRIVE 1
7665  030614    004                             .BYTE   4               ;DRIVE 2
7666  030615    010                             .BYTE   10              ;DRIVE 3
```

```
7667  030616  020                         .BYTE   20      ;DRIVE 4
7668  030617  040                         .BYTE   40      ;DRIVE 5
7669  030620  100                         .BYTE   100     ;DRIVE 6
7670  030621  200                         .BYTE   200     ;DRIVE 7
7671
7672                              ;*RP04 TO RH11 "MASS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
7673                              ;*CALLING IT FATAL (MCPEMX=1 WORD)
7674
7675  030622  000003               MCPEMX: .WORD   3
7676
7677                              ;*STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RP04),
7678                              ;*RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).
7679  030624  176700               RPADR:  .WORD   176700
7680  030626  000254  000240       RPVEC:  .WORD   254,5*32.
7681
7682                              ;*MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
7683  030632  000004               MXLACT: .WORD   4
7684                              ;*MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
7685  030634  001000               MXDLTA: .WORD   8.*64.
7686                              ;*MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
7687  030636  000200               MNDLTA: .WORD   2*64.
7688                              ;*MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
7689  030640  000005               MXWNDW: .WORD   5
7690
7691                              ;*DEFINITIONS OF THE RH11/RP04 ADDRESS INDEXES
7692
7693          000000               RHCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
7694          000002               RHWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
7695          000004               RHBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
7696          000006               RHDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
7697          000010               RHCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
7698          000012               RHDS1=12                ;DRIVE STATUS REGISTER (DRIVE REG 01)
7699          000014               RHER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
7700          000016               RHAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
7701          000020               RHLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
7702          000022               RHDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
7703          000024               RHMR=24                 ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
7704          000026               RHDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
7705          000030               RHSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
7706          000032               RHOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
7707          000034               RHCA=34                 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
7708          000036               RHCC=36                 ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
7709          000040               RHER2=40                ;ERROR REGISTER #2 (DRIVE REG. 14)
7710          000042               RHER3=42                ;ERROR REGISTER #3 (DRIVE REG. 15)
7711          000044               RHEC1=44                ;ECC POSITION REGISTER (DRIVE REG. 16)
7712          000046               RHEC2=46                ;ECC PATTERN REGISTER (DRIVE REG.  17)
7713
7714                              ;*RH11/RP04 DRIVER INIT. CODE
7715                                      ;*THIS ROUTINE WILL DETERMINE WHICH RP04 DRIVES ARE
7716                                      ;*AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
7717                                      ;*TO THE PROPER STATE FOR EACH DRIVE.
7718                                      ;*NOTE: THIS ROUTINE CALLS DRVINT
7719                                      ;
7720                              ;*CALL
```

```
7721                                        ;
7722                                        ;*      JSR     PC,RPINIT
7723                                        ;*      RETURN
7724                                        ;*
7725                                        ;*NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
7726                                        ;
7727  030642  004037  037014          RPINIT: JSR     R0,SAVR15       ;GO SAVE R1-R5
7728  030646  013746  177776                  MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
7729  030652  012737  000240  177776          MOV     #(5*32.),@#PS   ;CHANGE THE PRIORITY TO 5
7730  030660  004737  036552                  JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
7731  030664  012701  030446                  MOV     #RPERRS,R1      ;FIRST ADDRESS TO BE CLEARED
7732  030670  012702  030566                  MOV     #SEEKFG,R2      ;LAST ADDRESS TO BE CLEARED
7733  030674  005021                    1$:   CLR     (R1)+           ;CLEAR
7734  030676  020102                          CMP     R1,R2           ;ARE WE DONE?
7735  030700  101775                          BLOS    1$              ;BRANCH IF NO
7736  030702  012702  030610                  MOV     #DTUW,R2        ;LAST ADDRESS
7737  030706  012721  177777          2$:     MOV     #-1,(R1)+       ;INITIALIZE
7738  030712  020102                          CMP     R1,R2           ;DONE?
7739  030714  101774                          BLOS    2$              ;LOOP IF NO
7740  030716  012737  177777  030466          MOV     #-1,DRVSTA      ;SET ALL DRIVES TO UNAVAILABLE
7741  030724  012737  177777  030470          MOV     #-1,DRVSTA+2
7742  030732  012737  177777  030472          MOV     #-1,DRVSTA+4
7743  030740  012737  177777  030474          MOV     #-1,DRVSTA+6
7744  030746  013703  030626                  MOV     RPVEC,R3        ;SETUP THE RH11/RPO4 VECTOR
7745  030752  012723  033274                  MOV     #ISR,(R3)+
7746  030756  013713  030630                  MOV     RPVEC+2,(R3)
7747  030762  013704  030624                  MOV     RPADR,R4        ;FIRST ADDRESS OF RH11/RPO4
7748  030766  012764  000040  000010          MOV     #BIT05,RHCS2(R4)  ;MASSBUS INIT
7749  030774  005001                          CLR     R1              ;START WITH DRIVE 0
7750  030776  004037  031070          3$:     JSR     R0,DRVINT       ;INIT THE DRIVE
7751  031002  031062                          B$
7752  031004  005201                    4$:   INC     R1              ;GO TO NEXT DRIVE
7753  031006  042701  177770                  BIC     #^C7,R1         ;MASK OUT UNUSED BITS
7754  031012  001371                          BNE     3$              ;BR IF MORE DRIVES TO GO
7755  031014  012701  000007                  MOV     #7,R1           ;START WITH DRIVE 7
7756  031020  005037  177776                  CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
7757  031024  105761  030516          5$:     TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
7758  031030  001405                          BEQ     7$              ;BR NOT WAITING
7759  031032  004737  036206                  JSR     PC,SET.IE       ;SET INTERRUPT
7760  031036  105761  030516          6$:     TSTB    DPINT(R1)       ;DRIVE SWITCHED PORTS ?
7761  031042  001375                          BNE     6$              ;BR IF NOT
7762  031044  005301                    7$:   DEC     R1              ;GO TO THE NEXT DRIVE
7763  031046  100366                          BPL     5$              ;CHECK NEXT DRIVE
7764  031050  012637  177776                  MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
7765  031054  004037  037034                  JSR     R0,GETR1S       ;RESTORE R1-R5
7766  031060  000207                          RTS     PC              ;BYE-BYE
7767  031062  105060  030466          8$:     CLRB    DRVSTA(R0)      ;SET DRIVE STATUS TO OFFLINE
7768  031066  000746                          BR      4$              ;CONTINUE WITH THE INIT
7769
7770                                        ;*DRIVE INIT. ROUTINE
7771                                        ;*THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
7772                                        ;*AN RPO4. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
7773                                        ;*IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
7774                                        ;*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     RPADR,R4        ;UNIBUS ADDRESS OF RH11/RP04 (RHCS1)
                                        ;*      JSR     R0,DRVINT       ;CALLED BY A JSR
                                        ;*      RETURN1                 ;ERROR OCCURRED (PARITY)
                                        ;*      RETURN2                 ;NORMAL RETURN
                                        ;
031070  004037  037014          DRVINT: JSR     R0,SAVR15       ;SAVE R1-R5
031074  112761  177777  030466          MOVB    #-1,DRVSTA(R1)  ;START DRIVE STATUS AS NONEXISTENT
031102  010103                          MOV     R1,R3           ;COPY THE DRIVE NUMBER INTO
031104  006303                          ASL     R3              ;R3 AND POSITION IT FOR TABLE INDEXING
031106  005063  030476                  CLR     DRVTYP(R3)      ;CLEAR THE DRIVE TYPE INDICATOR
031112  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT A DRIVE
031116  112714  000111                  MOVB    #111,(R4)       ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
031122  032764  010000  000010          BIT     #BIT12,RHCS2(R4) ;NONEXISTENT DRIVE?
031130  001403                          BEQ     1$              ;NO---BRANCH
031132  004737  036206                  JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
031136  000465                          BR      6$              ;LEAVE THIS ROUTINE
031140  032714  004000          1$:     BIT     #BIT11,(R4)     ;SEE IF DRIVE AVAILABLE
031144  001466                          BEQ     8$              ;BR IF DRIVE NOT AVAILABLE
031146  004037  035610                  JSR     R0,RD.RP        ;READ THE DRIVE TYPE REG.
031152  000026                          RHDT
031154  031314                          7$                      ;ERROR RETURN ADDRESS
031156  012605                          MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
031160  010563  030476                  MOV     R5,DRVTYP(R3)   ;SAVE THE DRIVE TYPE
031164  022705  020020                  CMP     #20020,R5       ;IS IT A SINGLE PORT RP04?
031170  001403                          BEQ     2$              ;BRANCH IF YES
031172  022705  024020                  CMP     #24020,R5       ;IS IT A DUAL PORT RP04?
031176  001045                          BNE     6$              ;BRANCH IF NOT
031200  012746  000121          2$:     MOV     #121,-(SP)      ;DO A "READ-IN PRESET"
031204  004037  035746                  JSR     R0,WRT.RP
031210  000000                          RHCS1
031212  031314                          7$
031214  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT22=1
031220  004037  035746                  JSR     R0,WRT.RP
031224  000032                          RHOF
031226  031314                          7$
031230  004037  035610                  JSR     R0,RD.RP        ;READ RHDS1
031234  000012                          RHDS1
031236  031314                          7$
031240  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
031242  100011                          BPL     4$              ;BRANCH IF ATA=0
031244  116164  030612  000016          MOVB    ATABIT(R1),RHAS(R4) ;CLEAR ATTENTION BIT
031252  004037  035610                  JSR     R0,RD.RP        ;FIND OUT WHY ATA=1
031256  000014                          RHER1
031260  031314                          7$
031262  006126                          ROL     (SP)+           ;IS IT UNSAFE?
031264  100412                          BMI     6$              ;BRANCH IF YES
031266  005105                  4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
031270  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
031274  001004                          BNE     5$              ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
031276  112761  000001  030466          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
031304  000402                          BR      6$
```

```
 7829  031306  105061  030466          5$:     CLRB    DRVSTA(R1)      ;DRIVE STATUS = OFFLINE
 7830  031312  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
 7831  031314  004037  037034          7$:     JSR     R0,GETR15       ;RESTORE R1-R5
 7832  031320  000200                          RTS     R0              ;RETURN
 7833  031322  012763  003720  030570  8$:     MOV     #2000.,TIMER(R3)  ;START 2 SEC TIMER
 7834  031330  012764  000000  000012          MOV     #0,RHDS1(R4)    ;SET PORT REQUEST
 7835  031336  105061  030466                  CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
 7836  031342  112761  177777  030516          MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INIDICATOR
 7837  031350  000760                          BR      6$              ;EXIT
 7838
 7839                                  ;*REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
 7840                                  ;
 7841                                  ;*CALL
 7842                                  ;
 7843                                  ;*      JSR     R0,@#RP04       ;CALL THE RP04 DRIVER
 7844                                  ;*      PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
 7845                                  ;*      RETURN1                 ;RETURN HERE IF QUEUE IS FULL
 7846                                  ;*      RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
 7847                                                                  ;IS AN ERROR CONDITION
 7848
 7849  031352  013746  177776          RP04:   MOV     @#PS,-(SP)      ;SAVE THE CALLING STATUS
 7850  031356  013737  030630  177776          MOV     RPVEC+2,@#PS    ;DON'T ALLOW ANY RP04 INTERRUPTS
 7851  031364  112737  000001  030542          MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
 7852  031372  004037  037014                  JSR     R0,SAVR15       ;SAVE R1-R5
 7853  031376  012002                          MOV     (R0)+,R2        ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
 7854  031400  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
 7855  031404  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
 7856  031406  013704  030624                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF RHCS1
 7857  031412  105761  030466                  TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
 7858  031416  003021                          BGT     1$              ;BRANCH IF ONLINE
 7859  031420  105761  030544                  TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
 7860  031424  001043                          BNE     3$              ;BRANCH IF YES
 7861  031426  105761  030516                  TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
 7862  031432  001056                          BNE     7$              ;BR IF YES
 7863  031434  013704  030624                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF RHCS1
 7864  031440  004037  031070                  JSR     R0,DRVINT       ;GO INIT. THE DRIVE
 7865  031444  000446                          BR      6$              ;ERROR RETURN
 7866  031446  105761  030516                  TSTB    DPINT(R1)       ;SEE IF PORT REQEST OUTSTANDING
 7867  031452  001046                          BNE     7$              ;BR IF IT IS
 7868  031454  105761  030466                  TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
 7869  031460  003454                          BLE     8$              ;BR IF NOT
 7870  031462  105761  030526          1$:     TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
 7871  031466  001040                          BNE     7$              ;BR IF YES
 7872  031470  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT THE DRIVE
 7873  031474  004037  036654                  JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
 7874  031500  000421                          BR      5$              ;QUEUE IS FULL
 7875  031502  122762  000103  000002          CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
 7876  031510  001003                          BNE     2$              ;BR IF NO
 7877  031512  112761  177777  030544          MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
 7878  031520  105761  030456          2$:     TSTB    DRVACT(R1)      ;IS THIS DRIVE ACTIVE?
 7879  031524  001006                          BNE     4$              ;BR IF YES
 7880  031526  004737  031640                  JSR     PC,OPT          ;CALL THE OPTIMIZER
 7881  031532  000403                          BR      4$
 7882  031534  052762  120000  000016  3$:     BIS     #BIT15!BIT13,16(R2)     ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
```

```
7883  031542  005720                    4$:     TST     (R0)+
7884  031544  004037  037034            5$:     JSR     R0,GETR15       ;RESTORE R1-R5
7885  031550  105037  030542                    CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
7886  031554  012637  177776                    MOV     (SP)+,@#PS      ;RETURN "PS" TO USER LEVEL
7887  031560  000200                            RTS     R0              ;RETURN TO CALLER
7888  031562  004737  032700            6$:     JSR     PC,CI7          ;GO HANDLE THE PARITY ERROR
7889  031566  000765                            BR      4$
7890  031570  004037  036654            7$:     JSR     R0,DRVQUE       ;PUT REQUEST IN QUEUE
7891  031574  000763                            BR      5$              ;QUEUE IS FULL
7892  031576  032714  000100                    BIT     #BIT06,(R4)     ;IS 'IE' SET ALREADY ?
7893  031602  001357                            BNE     4$              ;BR IF IT IS
7894  031604  004737  036206                    JSR     PC,SET.IE       ;SET INTERRUPT
7895  031610  000754                            BR      4$              ;RETURN, REQUEST IN QUEUE
7896  031612  105761  030466            8$:     TSTB    DRVSTA(R1)      ;SEE IF DRIVE OFFLINE OR UNSAFE
7897  031616  002404                            BLT     9$              ;BR IF UNSAFE
7898  031620  052762  140000  000016            BIS     #BIT15!BIT14,16(R2)  ;ERROR--DRIVE IS OFFLINE
7899  031626  000745                            BR      4$              ;GO TO EXIT
7900  031630  052762  110000  000016  9$:       BIS     #BIT15!BIT12,16(R2)  ;ERROR--DRIVE IS UNSAFE
7901  031636  000741                            BR      4$              ;GO TO EXIT
7902
7903                                    ;*OPTIMIZER-CALLED FOR A PARTICULAR DRIV
7904                                    ;
7905                                    ;*CALL
7906                                    ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
7907                                    ;*      JSR     PC,OPT          ;SETUP A COMMAND
7908                                    ;
7909  031640  004037  037014            OPT:    JSR     R0,SAVR15       ;SAVE R1-R5
7910  031644  013746  177776                    MOV     @#PS,-(SP)      ;SAVE PROC. STATUS
7911  031650  146137  030612  030540            BICB    ATABIT(R1),SRCHWT       ;CLEAR "SEARCH WAIT" KEY
7912  031656  004737  036730                    JSR     PC,GETREQ       ;GET "DPB" POINTER OF REQUEST
7913  031662  005702                            TST     R2              ;IS THERE A REQUEST IN QUEUE?
7914  031664  001452                            BEQ     6$              ;NO--BRANCH TO EXIT
7915  031666  105761  030466                    TSTB    DRVSTA(R1)      ;IS DRIVE ONLINE?
7916  031672  003006                            BGT     1$              ;YES--BRANCH
7917  031674  004737  036752                    JSR     PC,POPQUE       ;NO--REMOVE REQUEST FROM QUEUE
7918  031700  052762  140000  000016            BIS     #BIT15!BIT14,16(R2)  ;SET ERROR BIT OF STATUS/ERROR INDICATOR
7919  031706  000434                            BR      5$              ;BRANCH TO EXIT
7920  031710  012764  000000  000012  1$:       MOV     #0,RHDS1(R4)    ;SEIZE THE DRIVE OR SET REQUEST
7921  031716  032714  004000                    BIT     #BIT11,(R4)     ;DRIVE AVAILABLE ?
7922  031722  001441                            BEQ     7$              ;BR IF NOT
7923  031724  122762  000150  000002            CMPB    #150,2(R2)      ;IS THE REQUEST FOR I/O?
7924  031732  002403                            BLT     2$              ;YES--BRANCH
7925  031734  004737  032264                    JSR     PC,CI4          ;CALL THE COMMAND INITIATOR
7926  031740  000417                            BR      5$              ;BRANCH TO EXIT
7927  031742  005737  030610            2$:     TST     DTUW            ;DATA TRANSFER UNDERWAY?
7928  031746  002012                            BGE     4$              ;YES--GO START A SEARCH
7929  031750  005737  030566                    TST     SEEKFG          ;DO IMPLIED SEEKS?
7930  031754  100404                            BMI     3$              ;YES---BRANCH
7931  031756  004037  033136                    JSR     R0,LA           ;NO--DO LOOK AHEAD
7932  031762  000406                            BR      5$              ;RETURN HERE ON A PARITY ERROR
7933  031764  000403                            BR      4$              ;GO START A SEARCH
7934  031766  004737  032050            3$:     JSR     PC,CI1          ;START A DATA TRANSFER
7935  031772  000402                            BR      5$
7936  031774  004737  032156            4$:     JSR     PC,CI3          ;START A SEARCH
```

```
7937  032000  012637  177776          5$:     MOV     (SP)+,@#PS      ;RESTORE PROC. STATUS
7938  032004  004037  037034                  JSR     R0,GETRIS       ;RESTORE R1-R5
7939  032010  000207                          RTS     PC
7940  032012  032714  000100          6$:     BIT     #BIT06,(R4)     ;SEE IF 'IE' ALREADY SET
7941  032016  001370                          BNE     5$              ;BR IF SET
7942  032020  004737  036206                  JSR     PC,SET.IE       ;SET "IE" WITHOUT A "TRE"
7943  032024  000765                          BR      5$              ;EXIT THE ROUTINE
7944  032026  112761  177777  030526  7$:     MOVB    #-1,DPRQS(R1)   ;SET PORT REQUEST INDICATOR
7945  032034  010103                          MOV     R1,R3           ;SET UP TO ADDRESS WORDS
7946  032036  006303                          ASL     R3              ;CONVERT TO WORD INDEX
7947  032040  012763  023420  030570          MOV     #10000.,TIMER(R3) ;START 10 SEC TIMER
7948  032046  000761                          BR      6$              ;SET 'IE'
7949
7950                                  ;*COMMAND INITIATOR
7951                                  ;
7952                                  ;*CALL
7953                                  ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER
7954                                  ;*      MOV     #DPB,R2         ;ADDRESS OF DPB
7955                                  ;*      JSR     PC,CI?          ;CI?= CI1,CI3, OR CI4
7956                                  ;*                              ;WHERE:
7957                                  ;*                              ;CI1=DATA TRANSFER
7958                                  ;*                              ;CI2=SEARCH REQUESTED BY DATA XFER
7959                                  ;*                              ;CI4=NOT DATA TRANSFER
7960
7961  032050  004737  036752          CI1:    JSR     PC,POPQUE       ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
7962  032054  010237  030536                  MOV     R2,TRNSWT       ;PUT REQ. IN TRANSFER WAIT QUEUE
7963  032060  010203                  CI2:    MOV     R2,R3           ;DPB ADDRESS TO R3
7964  032062  013704  030624                  MOV     RPADR,R4        ;RHCS1 ADDRESS
7965  032066  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT DRIVE
7966  032072  062703  000004                  ADD     #4,R3           ;DESIRED WORD COUNT
7967  032076  062704  000002                  ADD     #2,R4           ;RHWC ADDRESS
7968  032102  012324                          MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
7969  032104  012324                          MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
7970  032106  012346                          MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
7971  032110  004037  035746                  JSR     R0,WRT.RP       ;CALL THE LOAD(WRITE) ROUTINE
7972  032114  000006                          RHDA                    ;INDEX OF REGISTER TO LOAD
7973  032116  032700                          CI7                     ;ERROR RETURN ADDRESS
7974  032120  012346                          MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
7975  032122  004037  035746                  JSR     R0,WRT.RP
7976  032126  000034                          RHCA
7977  032130  032700                          CI7
7978  032132  016246  000002                  MOV     2(R2),-(SP)     ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
7979  032136  004037  035746                  JSR     R0,WRT.RP
7980  032142  000000                          RHCS1
7981  032144  032700                          CI7
7982  032146  010137  030610                  MOV     R1,DTUW         ;SET "DATA TRANSFER UNDERWAY"
7983  032152  000137  032642                  JMP     CIS
7984  032156  013704  030624          CI3:    MOV     RPADR,R4        ;RHCS1 ADDRESS
7985  032162  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT DRIVE
7986  032166  016246  000012                  MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
7987  032172  004037  035746                  JSR     R0,WRT.RP
7988  032176  000034                          RHCA
7989  032200  032700                          CI7
7990  032202  116203  000010                  MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
```

```
7991  032206  163703  030640                  SUB     MXWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
7992  032212  002002                          BGE     1$
7993  032214  062703  000026                  ADD     #22.,R3
7994  032220  010346                  1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
7995  032222  116266  000011  000001          MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
7996  032230  004037  035746                  JSR     R0,WRT.RP       ;LOAD DESIRED TRACK & SECTOR
7997  032234  000006                          RHDA
7998  032236  032700                          CI7
7999  032240  012746  000131                  MOV     #131,-(SP)      ;START A SEARCH
8000  032244  004037  035746                  JSR     R0,WRT.RP
8001  032250  000000                          RHCS1
8002  032252  032700                          CI7
8003  032254  156137  030612  030540          BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
8004  032262  000567                          BR      CI5
8005  032264  013704  030624          CI4:    MOV     RPADR,R4        ;RHCS1 ADDRESS
8006  032270  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT DRIVE
8007  032274  116203  000002                  MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
8008  032300  122703  000131                  CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
8009  032304  001007                          BNE     1$              ;BRANCH IF NO
8010  032306  016246  000010                  MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
8011  032312  004037  035746                  JSR     R0,WRT.RP
8012  032316  000006                          RHDA
8013  032320  032700                          CI7
8014  032322  000403                          BR      2$              ;GO LOAD CYLINDER
8015  032324  122703  000105          1$:     CMPB    #105,R3         ;IS IT A SEEK COMMAND
8016  032330  001007                          BNE     3$              ;BRANCH IF NO
8017  032332  016246  000012          2$:     MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
8018  032336  004037  035746                  JSR     R0,WRT.RP
8019  032342  000034                          RHCA
8020  032344  032700                          CI7
8021  032346  000546                          BR      CI6
8022  032350  122703  000115          3$:     CMPB    #115,R3         ;IS IT AN "OFFSET" COMMAND?
8023  032354  001013                          BNE     4$              ;BR IF NO
8024  032356  004037  035610                  JSR     R0,RD.RP        ;MERGE THE OFFSET VALUE INTO RHOF
8025  032362  000032                          RHOF                    ;BUT DON'T CHANGE THE UPPER
8026  032364  032700                          CI7
8027  032366  116216  000001                  MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
8028  032372  004037  035746                  JSR     R0,WRT.RP       ;REGISTER (RHOF)
8029  032376  000032                          RHOF
8030  032400  032700                          CI7
8031  032402  000530                          BR      CI6             ;GO START THE COMMAND
8032  032404  122703  000107          4$:     CMPB    #107,R3         ;IS IT A "RECALIBRATE" COMMAND?
8033  032410  001525                          BEQ     CI6             ;BRANCH IF YES
8034  032412  122703  000117                  CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
8035  032416  001522                          BEQ     CI6             ;BRANCH IF YES
8036  032420  122703  000103                  CMPB    #103,R3         ;IS IT AN "UNLOAD" COMMAND?
8037  032424  001016                          BNE     5$              ;BRANCH IF NO
8038  032426  112761  000001  030456          MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
8039  032434  105061  030466                  CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
8040  032440  112761  000001  030544          MOVB    #1,ULDFLG(R1)   ;SET "UNLOAD IN PROGRESS" FLAG
8041  032446  010346                          MOV     R3,-(SP)        ;START THE "UNLOAD" COMMAND
8042  032450  004037  035746                  JSR     R0,WRT.RP
8043  032454  000000                          RHCS1
8044  032456  032700                          CI7
```

```
8045  032460  000207                          RTS     PC              ;RETURN TO USER
8046  032462  122703  000143          5$:     CMPB    #143,R3         ;IS IT A "SET FORMAT" COMMAND?
8047  032466  001014                          BNE     6$              ;BRANCH IF NO
8048  032470  004037  035610                  JSR     R0,RD.RP        ;READ THE OFFSET REGISTER
8049  032474  000032                          RHOF
8050  032476  032700                          CI7
8051  032500  116266  000001  000001          MOVB    1(R2),1(SP)     ;COMBINE "FMT22","ECI", AND "HCI"
8052  032506  004037  035746                  JSR     R0,WRT.RP       ;LOAD "FMT22", "ECI", AND/OR "HCI".
8053  032512  000032                          RHOF
8054  032514  032700                          CI7
8055  032516  000436                          BR      12$
8056  032520  122703  000141          6$:     CMPB    #141,R3         ;IS IT A "GET REGISTER" COMMAND?
8057  032524  001023                          BNE     10$             ;BRANCH IF NO
8058  032526  016203  000006          7$:     MOV     6(R2),R3        ;POINTS TO 1ST ADDRESS OF WHERE
8059                                                                  ;TO PUT THE REGISTER(S)
8060  032532  116237  000010  032550          MOVB    10(R2),9$       ;INIT. THE INDEX FOR THE FIRST REG.
8061  032540  116205  000011                  MOVB    11(R2),R5       ;INDEX OF LAST REG. TO MOVE
8062  032544  004037  035610          8$:     JSR     R0,RD.RP        ;READ RPO4 REGISTER
8063  032550  000000          9$:     RHCS1                           ;INDEX OF REG. TO READ
8064  032552  032700                          CI7
8065  032554  012623                          MOV     (SP)+,(R3)+     ;GET THE CONTENTS OF RH11/RPO4 REG.
8066  032556  023705  032550                  CMP     9$,R5           ;LAST REG. BEEN READ?
8067  032562  001414                          BEQ     12$             ;GET OUT IF YES
8068  032564  062737  000002  032550          ADD     #2,9$           ;INCREASE THE INDEX BY 2
8069  032572  000764                          BR      8$              ;LOOP--MORE TO READ
8070  032574  122703  000145          10$:    CMPB    #145,R3         ;IS IT A "SELECT DRIVE" COMMAND?
8071  032600  001405                          BEQ     12$             ;BRANCH IF YES
8072  032602  010346                  11$:    MOV     R3,-(SP)        ;LOAD THE COMMAND
8073  032604  004037  035746                  JSR     R0,WRT.RP
8074  032610  000000                          RHCS1
8075  032612  032700                          CI7
8076  032614  004737  036752          12$:    JSR     PC,POPQUE       ;REMOVE REQ. FROM QUEUE
8077  032620  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET THE "DONE" BIT
8078  032626  005737  030564                  TST     SAVEFG          ;SAVE THE RH11/RPO4 REGISTERS?
8079  032632  100002                          BPL     13$             ;BRANCH IF NO
8080  032634  004737  036110                  JSR     PC,SVRH11       ;YES--GO SAVE THE REGISTERS
8081  032640  000207                  13$:    RTS     PC              ;RETURN TO USER
8082  032642  006301                  CI5:    ASL     R1
8083  032644  012761  001750  030570          MOV     #1000.,TIMER(R1)        ;SET A ONE SECOND TIMER
8084  032652  006201                          ASR     R1
8085  032654  112761  000001  030456          MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE
8086  032662  000207                          RTS     PC              ;RETURN TO THE USER
8087  032664  010346                  CI6:    MOV     R3,-(SP)        ;LOAD THE COMMAND
8088  032666  004037  035746                  JSR     R0,WRT.RP
8089  032672  000000                          RHCS1
8090  032674  032700                          CI7
8091  032676  000761                          BR      CI5
8092  032700  005702                  CI7:    TST     R2              ;ANYTHING IN QUEUE ?
8093  032702  001405                          BEQ     CI7B            ;BR IF NOT
8094  032704  012762  104000  000016          MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
8095  032712  004737  036110                  JSR     PC,SVRH11       ;GO SAVE THE RH11/RPO4 REGISTERS
8096  032716  012746  000111          CI7B:   MOV     #111,-(SP)      ;DO A "DRIVE CLEAR"
8097  032722  004037  035746                  JSR     R0,WRT.RP
8098  032726  000000                          RHCS1
```

```
8099  032730  032770                                  CIB
8100  032732  004737  036634                          JSR     PC,EMPTYQ       ;EMPTY THE QUEUE
8101  032736  105061  030544                          CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD IN QUEUE FLAG
8102  032742  105061  030456                          CLRB    DRVACT(R1)      ;DRIVE IS IDLE
8103  032746  020137  030610                          CMP     R1,DTUW         ;IF THIS DRIVE HAD AN I/O REQUEST
8104  032752  001005                                  BNE     1$              ;IN PROGRESS CLEAR ALL OF THE FLAGS
8105  032754  005037  030536                          CLR     TRNSWT
8106  032760  012737  177777  030610                  MOV     #-1,DTUW
8107  032766  000207                          1$:     RTS     PC
8108  032770  004037  037014                  CIB:    JSR     R0,SAVR15       ;SAVE R1-R5
8109  032774  013704  030624                          MOV     RPADR,R4        ;PICKUP THE ADDRESS OF THE FIRST REGISTER
8110  033000  005001                                  CLR     R1
8111  033002  005003                                  CLR     R3
8112  033004  105761  030456                  1$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
8113  033010  001423                                  BEQ     4$              ;BRANCH IF NO
8114  033012  013702  030536                          MOV     TRNSWT,R2       ;GET THE "TRANSFER WAIT" QUEUE
8115  033016  020137  030610                          CMP     R1,DTUW         ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
8116  033022  001402                                  BEQ     2$              ;BRANCH IF YES
8117  033024  004737  036730                          JSR     PC,GETREQ       ;GET THE DPB POINTER
8118  033030  005702                          2$:     TST     R2              ;QUEUE ENTRY FOR DRIVE ?
8119  033032  001405                                  BEQ     3$              ;BR IF NOT
8120  033034  052762  102000  000016                  BIS     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
8121  033042  004737  036110                          JSR     PC,SVRH11       ;SAVE RH11/RP04 REGISTERS
8122  033046  012763  177777  030570  3$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
8123  033054  105061  030456                          CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
8124  033060  105061  030544                  4$:     CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
8125  033064  005201                                  INC     R1              ;MOVE TO THE NEXT DRIVE
8126  033066  062703  000002                          ADD     #2,R3
8127  033072  042701  177770                          BIC     #^C7,R1
8128  033076  001342                                  BNE     1$              ;BRANCH IF MORE DRIVES
8129  033100  012737  177777  030610                  MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
8130  033106  005037  030536                          CLR     TRNSWT          ;CLEAR THE "TRANSFER WAIT" QUEUE
8131  033112  004737  036552                          JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
8132  033116  012764  000040  000010                  MOV     #BIT05,RHCS2(R4) ;DO A MASSBUS INIT.
8133  033124  004737  036206                          JSR     PC,SET.IE       ;SET "IE" WITHOUT "TRE"
8134  033130  004037  037034                          JSR     R0,GETR15       ;RESTORE THE REGISTERS
8135  033134  000207                                  RTS     PC              ;RETURN
8136
8137                                  ;*LOCK AHEAD ROUTINE
8138                                  ;
8139                                  ;*CALL
8140                                  ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER
8141                                  ;*      MOV     #DPB,R2         ;POINT TO DPB
8142                                  ;*      JSR     R0,LA           ;GO CHECK THE WINDOW
8143                                  ;*      RETURN1                 ;ERROR RETURN
8144                                  ;*      RETURN2                 ;START A SEARCH
8145                                  ;*      RETURN3                 ;START A DATA TRANSFER
8146
8147  033136  013704  030624          LA:     MOV     RPADR,R4        ;GET RHCS1'S ADDRESS
8148  033142  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT DRIVE
8149  033146  004037  035610                  JSR     R0,RD.RP        ;READ CURRENT CYLINDER
8150  033152  000036                          RHCC
8151  033154  033266                          4$                      ;ERROR RETURN ADDRESS
8152  033156  022662  000012                  CMP     (SP)+,12(R2)    ;IS CURRENT CYLINDER=DESIRED
```

```
8153
8154  033162  001037                          BNE     3$              ;CYLINDER?
8155  033164  105261  030554                  INCB    LACNT(R1)       ;EXIT IF NO
8156  033170  126137  030554  030632          CMPB    LACNT(R1),MXLACT ;INCREMENT THE LOOK AHEAD COUNT
8157  033176  003026                          BGT     2$              ;EXCEED MAX?
8158  033200  116203  000010                  MOVB    10(R2),R3       ;BRANCH IF YES
8159  033204  000303                          SWAB    R3              ;GET DESIRED SECTOR ADDRESS AND
8160  033206  006203                          ASR     R3              ;MULT. BY 64--ALIGN WITH
8161  033210  006203                          ASR     R3              ;LOOK AHEAD REGISTER
8162  033212  012737  000340  177776          MOV     #340,@#PS       ;PRIORITY LEVEL "7"
8163  033220  004037  035610                  JSR     R0,RD.RP        ;READ LOOK AHEAD REGISTER
8164  033224  000020                          RHLA
8165  033226  033266                          4$
8166  033230  162603                          SUB     (SP)+,R3        ;CALCULATE THE DELTA
8167  033232  002002                          BGE     1$
8168  033234  062703  002600                  ADD     #<22.*64.>,R3   ;MAKE THE DELTA POSITIVE
8169  033240  023703  030634          1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
8170  033244  002406                          BLT     3$              ;IF IT IS WITHIN THE
8171  033246  023703  030636                  CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
8172  033252  002003                          BGE     3$              ;THE LOOK AHEAD COUNT
8173  033254  105061  030554          2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
8174  033260  005720                          TST     (R0)+
8175  033262  005720                  3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
8176  033264  000200                          RTS     R0
8177  033266  004737  032700          4$:     JSR     PC,CI7          ;PROCESS THE ERROR
8178  033272  000200                          RTS     R0              ;TAKE ERROR RETURN
8179
8180                                  ;*INTERRUPT SERVICE ROUTINE
8181
8182  033274  112737  000001  030542  ISR:    MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
8183  033302  004037  037014                  JSR     R0,SAVR15       ;SAVE R1-R5
8184  033306  013704  030624                  MOV     RPADR,R4        ;ADDRESS OF RHSCS1
8185  033312  013701  030610                  MOV     DTUW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
8186  033316  002403                          BLT     1$              ;BRANCH IF NO DATA TRANSFER UNDERWAY
8187  033320  004737  033344                  JSR     PC,TD           ;CALL TRANSFER DONE
8188  033324  000402                          BR      2$              ;EXIT
8189  033326  004737  033626          1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
8190  033332  004037  037034          2$:     JSR     R0,GETR15       ;RESTORE R1-R5
8191  033336  105037  030542                  CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
8192  033342  000002                          RTI                     ;RETURN
8193
8194                                  ;*TRANSFER DONE ROUTINE
8195
8196  033344  105061  030456          TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
8197  033350  012737  177777  030610          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
8198  033356  006301                          ASL     R1
8199  033360  012761  177777  030570          MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
8200  033366  006201                          ASR     R1
8201  033370  013702  030536                  MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
8202  033374  005037  030536                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
8203  033400  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET DONE
8204  033406  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT THE DRIVE
8205  033412  004037  035610                  JSR     R0,RD.RP        ;TRANSFER ERROR(TRE=1)?
8206  033416  000000                          RHCS1
```

```
8207  033420  032700                          CI7
8208  033422  006126                          ROL     (SP)+
8209  033424  100421                          BMI     3$              ;BR IF YES
8210  033426  005737  030564                  TST     SAVEFG          ;SAVE THE RH11/RP04 REGISTERS?
8211  033432  100002                          BPL     1$              ;BRANCH IF NO
8212  033434  004737  036110                  JSR     PC,SVRH11       ;YES--SAVE THE REGISTERS
8213  033440  004737  033520          1$:     JSR     PC,WC           ;SEE IF WRITE CHECK TO BE PUT IN QUEUE
8214  033444  004737  036730                  JSR     PC,GETREQ       ;GET DPB POINTER
8215  033450  005702                          TST     R2              ;ENTRY FOR DRIVE ?
8216  033452  001403                          BEQ     2$              ;BR IF NOT
8217  033454  004737  031640                  JSR     PC,OPT          ;CALL OPTIMIZER
8218  033460  000520                          BR      SC2             ;SPECIAL CONDITION (ENTRY #2)
8219  033462  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
8220  033466  000515                          BR      SC2
8221  033470  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
8222  033476  004737  036634                  JSR     PC,EMPTYQ       ;EMPTY THE "DRIVES WAIT" QUEUE
8223  033502  004737  036110                  JSR     PC,SVRH11       ;SAVE THE RH11/RP04 REGISTERS
8224  033506  012714  040111                  MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
8225  033512  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
8226  033516  000501                          BR      SC2             ;SPECIAL CONDITION (ENTRY #2)
8227
8228                                  ;*FORCED WRITE CHECK ROUTINE
8229
8230  033520  005737  001310          WC:     TST     AUTOCK          ;AUTOMATIC WRITE CHECKS ?
8231  033524  001437                          BEQ     2$              ;BR IF NOT
8232  033526  122762  000002  000030          CMPB    #2,$CODE(R2)    ;LAST OPERATION WRITE DATA ?
8233  033534  001404                          BEQ     1$              ;BR IF IT WAS
8234  033536  122762  000003  000030          CMPB    #3,$CODE(R2)    ;LAST OPERATION WRITE HEADER & DATA ?
8235  033544  001027                          BNE     2$              ;BR IF NOT
8236  033546  004037  036654          1$:     JSR     R0,DRVQUE       ;PUT THE OPERATION IN THE QUEUE
8237  033552  000424                          BR      2$              ;QUEUE IS FULL
8238  033554  005062  000016                  CLR     16(R2)          ;CLEAR 'DONE' BIT IN DPB
8239  033560  116262  000030  000032          MOVB    $CODE(R2),$PREVO(R2)    ;SAVE WRITE OPERATION CODE
8240  033566  016262  000012  000040          MOV     $CYL(R2),$PREVA+2(R2)   ;SAVE CYLINDER
8241  033574  016262  000010  000036          MOV     $SEC(R2),$PREVA(R2)   ;SAVE SECTOR AND TRACK ADDRESSES
8242  033602  142762  000002  000030          BICB    #2,$CODE(R2)    ;CHANGE WRITE TO CHECK
8243  033610  142762  000020  000002          BICB    #20,$COMND(R2)  ;CHANGE DRIVER CODE TO WRITE CHECK
8244  033616  152762  000010  000002          BISB    #10,$COMND(R2)  ;FINISH CHANGING CODE TO WRITE CHECK
8245  033624  000207                  2$:     RTS     PC              ;EXIT
8246
8247                                  ;*SPECIAL CONDITION ROUTINE
8248
8249  033626  005001                  SC:     CLR     R1              ;START WITH DRIVE 0
8250  033630  012702  000001                  MOV     #1,R2
8251  033634  105761  030466          1$:     TSTB    DRVSTA(R1)      ;NONEXISTENT?
8252  033640  002030                          BGE     SC2             ;NO--BRANCH
8253  033642  105761  030516                  TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
8254  033646  001025                          BNE     SC2             ;YES--BRANCH
8255  033650  005201                          INC     R1              ;YES--MOVE TO THE NEXT DRIVE
8256  033652  106302                          ASLB    R2              ;MORE DRIVES?
8257  033654  103367                          BCC     1$              ;YES--BRANCH
8258  033656  012764  000040  000010          MOV     #BIT5,RHCS2(R4) ;INIT THE SYSTEM
8259  033664  000410                          BR      SC1A            ;TAKE ERROR EXIT
9260  033666  012701  000010          SC1:    MOV     #8.,R1          ;DO EIGHT DRIVES
```

```
8261  033672  005301                    1$:     DEC     R1              ;NEXT DRIVE
8262  033674  002754                            BLT     SC              ;BRANCH IF OUT OF DRIVES
8263  033676  105761  030456                    TSTB    DRVACT(R1)      ;IS THIS DRIVE IDLE?
8264  033702  001373                            BNE     1$              ;BRANCH IF NO
8265  033704  104001                            ERROR   1               ;ILLEGAL INTERRUPT
8266  033706  112764  177777  000016    SC1A:   MOVB    #-1,RHAS(R4)    ;CLEAR ANY OUTSTANDING ATTN BITS
8267  033714  004737  036206                    JSR     PC,SET.IE       ;GO SET INTERRUPT ENABLE
8268  033720  000207                            RTS     PC
8269  033722  012701  000003            SC2:    MOV     #3,R1           ;READ RHAS UP TO THREE TIMES
8270  033726  116403  000016            1$:     MOVB    RHAS(R4),R3     ;READ "RHAS"
8271  033732  001011                            BNE     2$              ;BRANCH IF ANY ATA BITS = 1
8272  033734  005301                            DEC     R1              ;COUNT THIS READ
8273  033736  003373                            BGT     1$              ;LOOP IF MORE READS ALLOWED
8274  033740  004037  035610                    JSR     R0,RD.RP        ;READ CONTROL AND STATUS REGISTER
8275  033744  000000                            RHCS1
8276  033746  032770                            CIB
8277  033750  106126                            ROLB    (SP)+           ;IS "IE"=1?
8278  033752  100345                            BPL     SC1             ;NO--TAKE ERROR EXIT
8279  033754  000207                            RTS     PC              ;YES--RETURN
8280  033756  005046                    2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
8281  033760  110316                            MOVB    R3,(SP)         ;AN "ATA"=1
8282  033762  012703  000001                    MOV     #1,R3
8283  033766  005001                            CLR     R1
8284  033770  030316                    SC4:    BIT     R3,(SP)         ;ATA=1?
8285  033772  001005                            BNE     SC5             ;YES--BRANCH
8286  033774  005201                    SC3:    INC     R1              ;MOVE TO THE NEXT DRIVE
8287  033776  106303                            ASLB    R3
8288  034000  001373                            BNE     SC4             ;BRANCH IF MORE TO CHECK?
8289  034002  005726                            TST     (SP)+           ;CLEAN OFF THE STACK
8290  034004  000207                            RTS     PC              ;RETURN TO USER
8291  034006  105761  030516            SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
8292  034012  001402                            BEQ     .+6             ;BR IF NOT
8293  034014  000137  034675                    JMP     SC13            ;PROCESS THE DRIVE
8294  034020  105761  030526                    TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
8295  034024  001402                            BEQ     .+6             ;BR IF NOT
8296  034026  000137  034676                    JMP     SC13            ;START THE OUTSTANDING COMMAND
8297  034032  023701  030610                    CMP     DTUW,R1         ;IS THIS DRIVE SETUP FOR I/O?
8298  034036  001003                            BNE     1$              ;NO---BRANCH
8299  034040  005726                            TST     (SP)+           ;YES---CLEAN OFF THE STACK (RHAS)
8300  034042  000137  033344                    JMP     TD              ;JUMP TO 'TRANSFER DONE'
8301  034046  105761  030466            1$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
8302  034052  003041                            BGT     SC6             ;BRANCH IF ONLINE
8303  034054  105761  030544                    TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
8304  034060  003422                            BLE     2$              ;BRANCH IF NO
8305  034062  004737  036730                    JSP     PC,GETREQ       ;GET DPB POINTER
8306  034066  004737  036110                    JSR     PC,SVRH11       ;SAVE THE RH11/RP04 REGISTERS
8307  034072  004737  034626                    JSR     PC,SC12         ;SAVE RHDS1, RHER1, RHER2, AND RHER3
8308                                                                    ;ALSO DO A DRIVE INIT (DRVINT)
8309  034076  105761  030466                    TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
8310  034102  003413                            BLE     3$              ;NO---BRANCH
8311  034104  032737  040000  030446           BIT     #BIT14,RPERRS   ;WAS THERE AN ERROR?
8312  034112  001002                            BNE     .+6             ;BR IF ERROR
8313  034114  000137  034536                    JMP     SC11            ;NO ERROR
8314  034120  013705  030450                    MOV     RPERRS+2,R5     ;YES -- PICKUP RHER1 AND
```

```
8315  034124  000463                          BR      SC6A            ;GO PROCESS THE ERROR
8316  034126  004737  034626          2$:     JSR     PC,SC12         ;SAVE RHDS1, RHER1, RHER2, AND RHER3
8317                                          ;ALSO DO A DRVINT
8318  034132  116164  030612  000016  3$:     MOVB    ATABIT(R1),RHAS(R4)  ;CLEAR THE ATTENTION BIT
8319  034140  012714  000111                  MOV     #111,(R4)       ;ISSUE A 'DRIVE CLEAR'
8320  034144  012714  000113                  MOV     #113,(R4)       ;RELEASE THE DRIVE
8321  034150  011605                          MOV     (SP),R5         ;PICKUP (RHAS) BEFORE THE ERROR CALL
8322  034152  104005                          ERROR   5               ;REPORT THE ERROR
8323  034154  000707                          BR      SC3             ;GO CHECK FOR MORE ATA'S
8324  034156  105761  030456          SC6:    TSTB    DRVACT(R1)      ;CHECK THE DRIVE ACTIVE INDICATOR
8325  034162  001552                          BEQ     SC10            ;BR IF IDLE
8326  034164  006301                          ASL     R1
8327  034166  012761  177777  030570          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
8328  034174  006201                          ASR     R1
8329  034176  004737  036730                  JSR     PC,GETREQ       ;GET THE RPB POINTER FROM THE QUEUE
8330  034202  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT DRIVE
8331  034206  004037  035610                  JSR     R0,RD.RP        ;READ THE RP04'S STATUS REG.
8332  034212  000012                          RHDS1
8333  034214  034464                          SCB
8334  034216  011605                          MOV     (SP),R5         ;AND PUT IT IN R5
8335  034220  006126                          ROL     (SP)+           ;WAS THERE AN ERROR?
8336  034222  100407                          BMI     1$              ;BR IF ERROR
8337  034224  105761  030456                  TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
8338  034230  003142                          BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
8339  034232  052762  100210  000016          BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
8340  034240  000472                          BR      SC7
8341  034242  004037  035610          1$:     JSR     R0,RD.RP        ;READ ERROR REGISTER #1
8342  034246  000014                          RHER1
8343  034250  034464                          SCB
8344  034252  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
8345  034254  004737  036110                  JSR     PC,SVRH11       ;SAVE RH11/RP04 REGISTERS
8346  034260  012746  000111                  MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
8347  034264  004037  035746                  JSR     R0,WRT.RP
8348  034270  000000                          RHCS1
8349  034272  034464                          SCB
8350  034274  006105                  SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
8351  034276  100406                          BMI     1$              ;BRANCH IF YES
8352  034300  005702                          TST     R2              ;ANYTHING IN QUEUE ?
8353  034302  001403                          BEQ     .+10            ;BR IF NOT
8354  034304  052762  100240  000016          BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
8355  034312  000445                          BR      SC7
8356  034314  004037  035610          1$:     JSR     R0,RD.RP        ;READ DRIVE STATUS REG. #1
8357  034320  000012                          RHDS1
8358  034322  034464                          SCB
8359  034324  011605                          MOV     (SP),R5         ;SAVE RHDS1 IN R5
8360  034326  006126                          ROL     (SP)+           ;"ERR"=1?
8361  034330  100014                          BPL     2$              ;BR IF NO--UNSAFE CLEARED
8362  034332  112761  177777  030466          MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
8363  034340  004737  036110                  JSR     PC,SVRH11       ;SAVE RH11/RP04 REGISTERS
8364  034344  116164  030612  000016          MOVB    ATABIT(R1),RHAS(R4)  ;CLEAR ATTENTION BIT
8365  034352  052762  110000  000016          BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
8366  034360  000422                          BR      SC7
8367  034362  105705                  2$:     TSTB    R5              ;"DRY" =1?
8368  034364  100415                          BMI     3$              ;BRANCH IF YES
```

```
8369  034366  112761  177777  030456          MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
8370  034374  112761  000001  030466          MOVB    #1,DRVSTA(R1)   ;ONLINE
8371  034402  006301                          ASL     R1
8372  034404  012761  072460  030570          MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
8373  034412  006201                          ASR     R1
8374  034414  000137  033774                  JMP     SC3
8375  034420  052762  100220  000016  3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
8376  034426  105061  030456          SC7:    CLRB    DRVACT(R1)              ;DRIVE IS IDLE
8377  034432  004737  036634                  JSR     PC,EMPTYQ               ;DUMP THE QUEUE
8378  034436  012714  000113                  MOV     #113,(R4)               ;RELEASE THE DRIVE
8379  034442  105761  030544                  TSTB    ULDFLG(R1)              ;UNLOAD IN PROGRESS OR QUEUE?
8380  034446  001002                          BNE     .+6                     ;BR IF NO
8381  034450  000137  033774                  JMP     SC3                     ;NO UNLOAD
8382  034454  105061  030544                  CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
8383  034460  000137  033774                  JMP     SC3
8384  034464  005726                  SC8:    TST     (SP)+                   ;REMOVE (RHAS) FROM THE STACK
8385  034466  105761  030456                  TSTB    DRVACT(R1)              ;IS DRIVE IDLE?
8386  034472  001404                          BEQ     1$                      ;YES--BRANCH
8387  034474  004737  036730                  JSR     PC,GETREQ               ;GET DPB POINTER
8388  034500  000137  032700                  JMP     CI7                     ;PROCESS THE PARITY ERROR
8389  034504  000137  032716          1$:     JMP     CI7B                    ;PROCESS THE PARITY ERROR
8390  034510  011605                  SC10:   MOV     (SP),R5 ;PUT (RHAS) IN R5
8391  034512  004737  034626                  JSR     PC,SC12                 ;SAVE RHDS1, RHER1, RHER2, AND RHER3
8392  034516  116164  030612  000016          MOVB    ATABIT(R1),RHAS(R4)     ;CLEAR ATTENTION BIT
8393  034524  012714  000113                  MOV     #113,(R4)               ;ISSUE A RELEASE
8394  034530  104002                          ERROR   2                       ;REPORT ILLEGAL DRIVE INTERRUP*
8395  034532  000137  033774                  JMP     SC3                     ;GO CHECK FOR MORE ATA'S
8396  034536  105761  030544          SC11:   TSTB    ULDFLG(R1)              ;"UNLOAD IN PROGRESS"?
8397  034542  003402                          BLE     1$                      ;BRANCH IF NO
8398  034544  105061  030544                  CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
8399  034550  105061  030456          1$:     CLRB    DRVACT(R1)              ;SET DRIVE IDLE
8400  034554  136137  030612  030540          BITB    ATABIT(R1),SRCHWT               ;DOING A SEARCH OPERATION FOR
8401                                                                          ;AN I/O COMMAND?
8402  034562  001012                          BNE     2$                      ;BRANCH IF YES
8403  034564  004737  036752                  JSR     PC,POPQUE               ;REMOVE REQUEST FROM QUEUE
8404  034570  052762  000200  000016          BIS     #BIT07,16(R2)           ;SET "DONE" BIT
8405  034576  005737  030564                  TST     SAVEFG                  ;SAVE THE RH11/RP04 REGISTERS?
8406  034602  100002                          BPL     2$                      ;BRANCH IF NO
8407  034604  004737  036110                  JSR     PC,SVRH11               ;YES--SAVE ALL OF THE RH11/RP04 REG'S
8408  034610  116164  030612  000016  2$:     MOVB    ATABIT(R1),RHAS(R4)     ;CLEAR ATTENTION BIT
8409  034616  004737  031640                  JSR     PC,CPT                  ;START A REQUEST
8410  034622  000137  033774                  JMP     SC3
8411  034626  010164  000010          SC12:   MOV     R1,RHCS2(R4)    ;SELECT DRIVE
8412  034632  016437  000012  030446          MOV     RHDS1(R4),RPERRS        ;SAVE THE FOUR REGISTERS THAT
8413  034640  016437  000014  030450          MOV     RHER1(R4),RPERRS+2      ;WILL TELL US SOMETHING
8414  034646  016437  000040  030452          MOV     RHER2(R4),RPERRS+4
8415  034654  016437  000042  030454          MOV     RHER3(R4),RPERRS+6
8416  034662  004037  031070                  JSR     R0,DRVINT               ;INIT. THE STATE OF THE DRIVE
8417  034666  000401                          BR      1$                      ;TAKE ERROR EXIT
8418  034670  000207                          RTS     PC                      ;RETURN
8419  034672  005726                  1$:     TST     (SP)+                   ;POP PC OFF OF THE STACK
8420  034674  000673                          BR      SC8                     ;PROCESS THE PARITY ERROR
8421  034676  006301                  SC13:   ASL     R1                      ;SETUP TO ADDRESS WORDS
8422  034700  012761  177777  030570          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
```

```
8423  034706  006201                          ASR     R1              ;
8424  034710  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT THE DRIVE
8425  034714  116164  030612  000016          MOVB    ATABIT(R1),RHAS(R4)  ;CLEAR THE ATTENTION BIT
8426  034722  032714  004000                  BIT     #BIT11,(R4)     ;DRIVE AVAILABLE ?
8427  034726  001006                          BNE     1$              ;BR IF AVAILABLE
8428  034730  006301                          ASL     R1
8429  034732  012761  023420  030570          MOV     #10000.,TIMER(R1)  ;START 10 SEC TIMER AGAIN
8430  034740  006201                          ASR     R1
8431  034742  000433                          BR      3$              ;EXIT
8432  034744  105761  030516          1$:     TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
8433  034750  001424                          BEQ     2$              ;BR IF NOT
8434  034752  105061  030516                  CLRB    DPINT(R1)       ;CLEAR THE INIT INDICATOR
8435  034756  004037  031070                  JSR     R0,DRVINT       ;GO INIT THE DRIVE
8436  034762  000240                          NOP                     ;DUMMY PARITY ERROR RETURN
8437  034764  105761  030466                  TSTB    DRVSTA(R1)      ;DRIVE ONLINE ?
8438  034770  003014                          BGT     2$              ;BR IF YES -- START ORDER
8439  034772  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE
8440  034774  001416                          BEQ     3$              ;BR IF NOT
8441  034776  004737  036730                  JSR     PC,GETREQ       ;GET DPB ADDRESS
8442  035002  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM USER THAT DRIVE OFFLINE
8443  035010  004737  036110                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
8444  035014  004737  036634                  JSR     PC,EMPTYQ       ;EMPTY THE REQUEST QUEUE
8445  035020  000404                          BR      3$
8446  035022  105061  030526          2$:     CLRB    DPREQS(R1)      ;CLEAR THE PORT REQUEST INDICATOR
8447  035026  004737  031640                  JSR     PC,OPT          ;START THE PENDING REQUEST
8448  035032  000137  033774          3$:     JMP     SCJ             ;PROCESS OTHER DRIVES
8449
8450                                  ;*RP04 TIMER ROUTINE
8451                                  ;*CALL
8452                                  ;*      MOV     #TIME,-(SP)     ;ELAPSED TIME IN MILLISECONDS ON THE STACK
8453                                  ;<      JSR     R0,RPTMR        ;CALL RP04 TIME ROUTINE
8454
8455  035036  005737  030542          RPTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
8456  035042  001032                          BNE     4$              ;IF NON ZERO EXIT
8457  035044  112737  000001  030543          MOVB    #1,ACTSTR       ;SET "ACTSTR"
8458  035052  004037  037014                  JSR     R0,SAVR15       ;SAVE R1-R5
8459  035056  005001                          CLR     R1              ;START WITH DRIVE 0
8460  035060  005003                          CLR     R3
8461  035062  005763  030570          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
8462  035066  002407                          BLT     2$              ;BRANCH IF NO
8463  035070  166663  000016  030570          SUB     16(SP),TIMER(R3)  ;COUNT THE INTERVAL
8464  035076  003003                          BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
8465  035100  004037  035134                  JSR     R0,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
8466  035104  000405                          BR      3$              ;GO TO THE EXIT
8467  035106  005201                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
8468  035110  005723                          TST     (R3)+
8469  035112  022701  000010                  CMP     #8.,R1          ;OUT OF DRIVES?
8470  035116  003361                          BGT     1$              ;BRANCH IF NO
8471  035120  004037  037034          3$:     JSR     R0,GETR15       ;RESTORE R1-R5
8472  035124  105037  030543                  CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
8473  035130  012616                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
8474  035132  000200                          RTS     R0              ;RETURN
8475
8476                                  ;*SOFTWARE TIMEOUT ROUTINE
```

```
                                        ;*
                                        ;*CALL: STO
                                        ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                        ;*      JSR     R0,STO          ;CALL--DRVACT MUST BE NONZERO
                                        ;*      RETURN
                                        
035134  013746  177776          STO:    MOV     @#PS,-(SP)      ;SAVE THE PROCESSOR STATUS
035140  013737  030630  177776          MOV     RPVEC+2,@#PS    ;SET THE "PS" TO RP04 BR LEVEL
035146  004037  037014                  JSR     R0,SAVR15       ;SAVE R1-R5
035152  013704  030624                  MOV     RPADR,R4        ;GET ADDRESS OF "RHCS1"
035156  010164  000010                  MOV     R1,RHCS2(R4)    ;SELECT THE DRIVE
035162  004037  035610                  JSR     R0,RD.RP        ;READ "DRIVE STATUS REG"
035166  000012                          RHDS1
035170  035502                          STO5
035172  105726                          TSTB    (SP)+           ;IS "DRY"=1?
035174  100477                          BMI     STO2            ;BR IF YES
035176  105761  030516          STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
035202  001074                          BNE     STO2            ;BR IF YES
035204  105761  030526                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
035210  001071                          BNE     STO2            ;BR IF YES
035212  013702  030536                  MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
035216  020137  030610                  CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
035222  001402                          BEQ     1$              ;BRANCH IF YES
035224  004737  036730                  JSR     PC,GETREQ       ;GET DPB ADDRESS
035230  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
035236  004737  036110                  JSR     PC,SVRH11       ;SAVE RH11/RP04 REGISTERS
035242  012764  000040  000010          MOV     #BIT05,RHCS2(R4) ;"INIT" THE MASS BUS
035250  105061  030456                  CLRB    DRVACT(R1)      ;DRIVE IS IDLE
035254  105061  030544                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
035260  005001                          CLR     R1              ;START WITH DRIVE 0
035262  005003                          CLR     R3
035264  004037  031070          2$:     JSR     R0,DRVINT       ;INIT. THIS DRIVE
035270  000504                          BR      STO5            ;PARITY ERROR RETURN
035272  105761  030456                  TSTB    DRVACT(R1)      ;DRIVE IDLE BEFORE THE INIT.?
035276  001414                          BEQ     4$              ;YES--BRANCH
035300  013702  030536                  MOV     TRNSWT,R2       ;GET TRANSFER WAIT QUEUE
035304  023701  030610                  CMP     DTUW,R1         ;WAS THERE I/O ON THIS DRIVE?
035310  001402                          BEQ     3$              ;YES--BRANCH
035312  004737  036730                  JSR     PC,GETREQ       ;GET THE DPB POINTER FROM QUEUE
035316  052762  100400  000016  3$:     BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT.
035324  105061  030456                  CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE TO IDLE
035330  105061  030544          4$:     CLRB    ULDFLG(R1)      ;NO UNLOAD
035334  012763  177777  030570          MOV     #-1,TIMER(R3)   ;STOP THE TIMER
035342  005723                          TST     (R3)+           ;UPDATE THE INDEX
035344  005201                          INC     R1              ;INCREMENT THE DRIVE NUMBER
035346  022701  000010                  CMP     #8.,R1          ;LAST DRIVE BEEN CHECKED?
035352  003344                          BGT     2$              ;NO--LOOP
035354  012737  177777  030610          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
035362  005037  030536                  CLR     TRNSWT          ;CLEAR TRANSFER WAIT QUEUE
035366  004737  036552                  JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
035372  000436                          BR      STO4            ;EXIT
035374  116405  000016          STO2:   MOVB    RHAS(R4),R5     ;READ ATTENTION REG
035400  136105  030612                  BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
035404  001017                          BNE     STO3            ;YES--BRANCH
```

```
8531  035406  105761  030516                      TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
8532  035412  001036                              BNE     ST06            ;BR IF YES - DRIVE NOT ONLINE
8533  035414  105761  030526                      TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
8534  035420  001052                              BNE     ST07            ;BR IF YES - NO RESPONSE TO REQUEST
8535  035422  020137  030610                      CMP     R1,DTUW         ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
8536  035426  001263                              BNE     ST01            ;BR IF NO
8537  035430  004037  035610                      JSR     R0,RD.RP        ;YES--CHECK "RDY"
8538  035434  000000                              RHCS1
8539  035436  035502                              ST05
8540  035440  105726                              TSTB    (SP)+
8541  035442  100255                              BPL     ST01            ;BR IF "RDY"=0
8542  035444  005720                      ST03:   TST     (R0)+           ;ADJUST FOR THE PROPER RETURN
8543  035446  105761  030516                      TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
8544  035452  001003                              BNE     1$              ;BR IF INIT PENDING
8545  035454  105761  030526                      TSTB    DPRQS(R1)       ;PORT REQUEST PENDING ?
8546  035460  001403                              BEQ     ST04            ;BR IF NOT
8547  035462  012763  177777  030570      1$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
8548  035470  004037  037034              ST04:   JSR     R0,GETR15       ;RESTORE R1-R5
8549  035474  012637  177776                      MOV     (SP)+,@#PS      ;RESTORE PROCESSOR STATUS
8550  035500  000200                              RTS     R0              ;RETURN
8551  035502  004737  032770              ST05:   JSR     PC,CIB          ;GO HANDLE THE PARITY ERROR
8552  035506  000770                              BR      ST04
8553  035510  105061  030516              ST06:   CLRB    DPINT(R1)       ;CLEAR THE INITIALIZE INDICATOR
8554  035514  105061  030466                      CLRB    DRVSTA(R1)      ;SET UNIT OFFLINE
8555  035520  012763  177777  030570              MOV     #-1,TIMER(R3)   ;STOP THE TIMER
8556  035526  004737  036730                      JSR     PC,GETREQ       ;GET THE DPB ADDRESS
8557  035532  005702                              TST     R2              ;REQUEST IN QUEUE ?
8558  035534  001755                              BEQ     ST04            ;BR IF NOT
8559  035536  052762  140000  000016              BIS     #BIT15!BIT14,16(R2)  ;INFORM THE USER DRIVE NOT AVAILABLE
8560  035544  000414                              BR      ST08            ;FINISH
8561  035546  012763  177777  030570  ST07:       MOV     #-1,TIMER(R3)   ;STOP THE TIMER
8562  035554  105061  030526                      CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST INDICATOR
8563  035560  004737  036730                      JSR     PC,GETREQ       ;GET DPB ADDRESS
8564  035564  005702                              TST     R2              ;QUEUE ENTRY FOR DRIVE ?
8565  035566  001403                              BEQ     ST08            ;BR IF NONE
8566  035570  052762  100004  000016              BIS     #BIT15!BIT2,16(R2)  ;INFORM USER OF PORT REQUEST ERROR
8567  035576  004737  036634              ST09:   JSR     PC,EMPTYQ       ;CLEAR THE QUEUE FOR THE DRIVE
8568  035602  004737  036110                      JSR     PC,SVRH11       ;SAVE THE REGISTERS
8569  035606  000730                              BR      ST04            ;RETURN
8570
8571                                  ;*ROUTINE TO READ A RH11/RP04 REGISTER
8572                                  ;
8573                                  ;*CALL
8574                                  ;*      JSR     R0,RD.RP        ;GO READ A REGISTER
8575                                  ;*      INDEX                   ;REG. INDEX FROM BASE
8576                                  ;*      ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
8577                                  ;*                              ;AT THIS ADDRESS
8578                                  ;*      RETURN                  ;CONTENTS OF REG. IS ON THE STACK
8579                                  ;
8580  035610  013737  030622  035734  RD.RP:   MOV     MCPEMX,RD.RP2   ;MAX. RETRYS ALLOWED
8581  035616  011646                          MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
8582  035620  013737  030624  035634          MOV     RPADR,RD.ADR    ;FORM THE DESIRED ADDRESS
8583  035626  062037  035634                  ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
8584  035632  013727                  RD.RP1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RP04
```

```
 8585  035634  000000                    RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
 8586  035636  000000                    RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
 8587  035640  013766  035636  000002            MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
 8588  035646  017746  172752                    MOV     @RPADR,-(SP)    ;READ RHCS1
 8589  035652  032716  020000                    BIT     #BIT13,(SP)     ;DID MCPE SET?
 8590  035656  001002                            BNE     1$              ;BRANCH IF YES
 8591  035660  022620                            CMP     (SP)+,(RO)+     ;ADJUST FOR RETURN
 8592  035662  000200                            RTS     RO              ;RETURN IF NO
 8593  035664  104003                    1$:     ERROR   3               ;REPORT "MCPE" ERROR
 8594  035666  005737  030610                    TST     DTUW            ;DATA TRANSFER UNDERWAY?
 8595  035672  100405                            BMI     2$              ;NO--BRANCH
 8596  035674  032716  040000                    BIT     #BIT14,(SP)     ;NO--"TRE"=1?
 8597  035700  001402                            BEQ     2$              ;NO--BRANCH
 8598  035702  005726                            TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
 8599  035704  000415                            BR      RD.RP3          ;TAKE THE FATAL ERROR EXIT
 8600  035706  052716  040000            2$:     BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
 8601  035712  000316                            SWAB    (SP)            ;POSITION BEFORE WRITING
 8602  035714  013737  030624  035730            MOV     RPADR,3$        ;FORM ADDRESS OF HIGH BYTE
 8603  035722  005237  035730                    INC     3$
 8604  035726  112637                            MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RHCS1
 8605  035730  000000                    3$:     .WORD   0               ;ADDRESS STORAGE
 8606  035732  005327                            DEC     (PC)+           ;EXCEEDED MAX. RETRYS
 8607  035734  000003                    RD.RP2: .WORD   3
 8608  035736  002335                            BGE     RD.RP1          ;BRANCH IF NO
 8609  035740  011000                    RD.RP3: MOV     (RO),RO         ;FATAL ERROR EXIT
 8610  035742  012616                            MOV     (SP)+,(SP)
 8611  035744  000200                            RTS     RO
 8612
 8613                                    ;*ROUTINE TO WRITE A RH11/RP04 REGISTER
 8614                                    ;
 8615                                    ;*CALL
 8616                                    ;*      MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
 8617                                    ;*      JSR     RO,WRT.RP       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
 8618                                    ;*      INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
 8619                                    ;*      ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
 8620                                    ;*      RETURN                  ;ERROR FREE RETURN
 8621
 8622  035746  016637  000002  036034    WRT.RP: MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
 8623  035754  012616                            MOV     (SP)+,(SP)      ;ADJUST THE STACK
 8624  035756  012037  036036                    MOV     (RO)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
 8625  035762  001020                            BNE     2$              ;BRANCH IF NOT RHCS1
 8626  035764  122737  000150  036034            CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
 8627  035772  002414                            BLT     2$              ;YES--DON'T GET THE OLD A16&A17, & PSEL
 8628  035774  004037  035610                    JSR     RO,RD.RP        ;NO---COMBINE A16&A17, & PSEL WITH
 8629  036000  000000                            RHCS1                   ;THE COMMAND BEFORE SENDING IT TO
 8630  036002  036020                            1$                      ;THE RH11/RP04
 8631  036004  000316                            SWAB    (SP)
 8632  036006  042716  177770                    BIC     #^C7,(SP)
 8633  036012  112637  036035                    MOVB    (SP)+,WRT.WD+1
 8634  036016  000402                            BR      2$
 8635  036020  011000                    1$:     MOV     (RO),RO         ;TAKE THE ERROR EXIT
 8636  036022  000200                            RTS     RO
 8637  036024  063737  030624  036036  2$:     ADD     RPADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
 8638  036032  012737                            MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
```

```
 8639  036034  000000                    WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
 8640  036036  000000                    WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
 8641  036040  004037  035610                    JSR     R0,RD.RP        ;CHECK FOR PARITY ERROR ON WRITE
 8642  036044  000014                            RHER1
 8643  036046  036100                            2$
 8644  036050  032726  000010                    BIT     #BIT03,(SP)+
 8645  036054  001413                            BEQ     3$              ;BRANCH IF "PAR=0"
 8646  036056  016037  177776  036070            MOV     -2(R0),1$       ;PICKUP THE INDEX
 8647  036064  004037  035610                    JSR     R0,RD.RP        ;READ THE REG.
 8648  036070  000000                    1$:     .WORD   0               ;       REG. INDEX
 8649  036072  036100                            2$                      ;RETURN TO THIS ADDRESS ON ERROR
 8650  036074  104004                            ERROR   4               ;REPORT THE PARITY ON WRITE ERROR
 8651  036076  005726                            TST     (SP)+           ;CLEAN OFF THE STACK
 8652  036100  011000                    2$:     MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
 8653  036102  000200                            RTS     R0
 8654  036104  005720                    3$:     TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
 8655  036106  000200                            RTS     R0
 8656
 8657                                    ;*ROUTINE TO SAVE THE RH11/RP04 REGISTERS AS PER DPB+14
 8658                                    ;
 8659                                    ;*CALL
 8660                                    ;*      MOV     #DPBNUM,R2      ;DPB POINTER TO R2
 8661                                    ;*      JSR     PC,SVRH11       ;SAVE THE DRIVES REG'S
 8662
 8663  036110  004037  037014            SVRH11: JSR     R0,SAVR15       ;SAVE REG.'S R1-R5
 8664  036114  005702                            TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
 8665  036116  001430                            BEQ     4$              ;BR IF NONE
 8666  036120  013704  030624                    MOV     RPADR,R4
 8667  036124  111264  000010                    MOVB    (R2),RHCS2(R4)  ;SELECT DRIVE
 8668  036130  016202  000014                    MOV     14(R2),R2       ;GET THE ERROR TABLE POINTER
 8669  036134  001421                            BEQ     4$              ;EXIT IF 0
 8670  036136  005003                            CLR     R3              ;COUNTER & POINTER
 8671  036140  012705  000022                    MOV     #RHDB,R5        ;PROBLEM REGISTER
 8672  036144  020305                    1$:     CMP     R3,R5           ;REACHED RHDB?
 8673  036146  001005                            BNE     2$              ;BR IF NO
 8674  036150  105764  000010                    TSTB    RHCS2(R4)       ;CHECK "OR"
 8675  036154  100402                            BMI     2$              ;BRANCH IF RHDB CAN BE READ
 8676  036156  005022                            CLR     (R2)+           ;ELSE SAVE IT AS 0'S
 8677  036160  000403                            BR      3$
 8678  036162  010446                    2$:     MOV     R4,-(SP)        ;FORM RH11/RP04 ADDRESS THAT IS
 8679  036164  060316                            ADD     R3,(SP)         ;TO BE READ
 8680  036166  013622                            MOV     @(SP)+,(R2)+    ;AND READ IT
 8681  036170  005723                    3$:     TST     (R3)+           ;MOVE TO NEXT REG INDEX
 8682  036172  020327  000046                    CMP     R3,#RHEC2       ;DONE?
 8683  036176  003762                            BLE     1$              ;BRANCH IF NO
 8684  036200  004037  037034            4$:     JSR     R0,GETR15       ;GET REGISTERS R1-R5
 8685  036204  000207                            RTS     PC              ;RETURN TO USER
 8686
 8687                                    ;*ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
 8688                                    ;*CALL
 8689                                    ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 8690                                    ;*      JSR     PC,SET.IE       ;SET "IE"
 8691                                    ;*      RETURN
 8692
```

```
 8693  036206  010446                    SET.IE: MOV     R4,-(SP)          ;SAVE R4
 8694  036210  013704  030624                    MOV     RPADR,R4          ;PICKUP ADDRESS OF RHCS1
 8695  036214  010164  000010                    MOV     R1,RHCS2(R4)      ;SELECT DRIVE
 8696  036220  011446                            MOV     (R4),-(SP)        ;READ RHCS1
 8697  036222  052716  040000                    BIS     #BIT14,(SP)       ;SET THE "TRE" BIT OF THE WORD READ
 8698  036226  000316                            SWAB    (SP)              ;ADJUST FOR DATO
 8699  036230  112714  000100                    MOVB    #BIT06,(R4)       ;SET "IE"
 8700  036234  032764  010000  000010            BIT     #BIT12,RHCS2(R4)  ;IS "NED"=1?
 8701  036242  001002                            BNE     1$                ;YES--CLEAR "TRE"
 8702  036244  005726                            TST     (SP)+             ;CLEAN OFF THE STACK
 8703  036246  000402                            BR      2$
 8704  036250  112664  000001            1$:     MOVB    (SP)+,1(R4)       ;CLEAR "TRE"
 8705  036254  012604                    2$:     MOV     (SP)+,R4          ;RESTORE R4
 8706  036256  000207                            RTS     PC                ;RETURN TO CALLER
 8707
 8708                                    ;*QUEUE COUNT
 8709  036260      000                   QCNT:   .BYTE   0                 ;DRIVE 0
 8710  036261      000                           .BYTE   0                 ;DRIVE 1
 8711  036262      000                           .BYTE   0                 ;DRIVE 2
 8712  036263      000                           .BYTE   0                 ;DRIVE 3
 8713  036264      000                           .BYTE   0                 ;DRIVE 4
 8714  036265      000                           .BYTE   0                 ;DRIVE 5
 8715  036266      000                           .BYTE   0                 ;DRIVE 6
 8716  036267      000                           .BYTE   0                 ;DRIVE 7
 8717
 8718                                    ;QUEUE INPUT POINTERS
 8719
 8720  036270  036352                    QINPT:  .WORD   QDRV0              ;DRIVE 0
 8721  036272  036372                            .WORD   QDRV1             ;DRIVE 1
 8722  036274  036412                            .WORD   QDRV2             ;DRIVE 2
 8723  036276  036432                            .WORD   QDRV3             ;DRIVE 3
 8724  036300  036452                            .WORD   QDRV4             ;DRIVE 4
 8725  036302  036472                            .WORD   QDRV5             ;DRIVE 5
 8726  036304  036512                            .WORD   QDRV6             ;DRIVE 6
 8727  036306  036532                            .WORD   QDRV7             ;DRIVE 7
 8728
 8729                                    ;QUEUE OUTPUT POINTERS
 8730
 8731  036310  036352                    QOUTPT: .WORD   QDRV0              ;DRIVE 0
 8732  036312  036372                            .WORD   QDRV1             ;DRIVE 1
 8733  036314  036412                            .WORD   QDRV2             ;DRIVE 2
 8734  036316  036432                            .WORD   QDRV3             ;DRIVE 3
 8735  036320  036452                            .WORD   QDRV4             ;DRIVE 4
 8736  036322  036472                            .WORD   QDRV5             ;DRIVE 5
 8737  036324  036512                            .WORD   QDRV6             ;DRIVE 6
 8738  036326  036532                            .WORD   QDRV7             ;DRIVE 7
 8739
 8740  036330  036352                    QSTART: .WORD   QDRV0              ;DRIVE 0 START ADDRESS
 8741  036332  036372                    QSTOP:  .WORD   QDRV1              ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
 8742  036334  036412                            .WORD   QDRV2             ;STOP DRIVE 1--START DRIVE 2
 8743  036336  036432                            .WORD   QDRV3             ;STOP DRIVE 2--START DRIVE 3
 8744  036340  036452                            .WORD   QDRV4             ;STOP DRIVE 3--START DRIVE 4
 8745  036342  036472                            .WORD   QDRV5             ;STOP DRIVE 4--START DRIVE 5
 8746  036344  036512                            .WORD   QDRV6             ;STOP DRIVE 5--START DRIVE 6
```

```
8747  036346  036532                          .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
8748  036350  036552                          .WORD   QTERM           ;STOP DRIVE 7
8749
8750                                  ;DRIVE REQUEST QUEUES
8751
8752  036352  000010                  QDRV0:  .BLKW   10
8753  036372  000010                  QDRV1:  .BLKW   10
8754  036412  000010                  QDRV2:  .BLKW   10
8755  036432  000010                  QDRV3:  .BLKW   10
8756  036452  000010                  QDRV4:  .BLKW   10
8757  036472  000010                  QDRV5:  .BLKW   10
8758  036512  000010                  QDRV6:  .BLKW   10
8759  036532  000010                  QDRV7:  .BLKW   10
8760          036552                  QTERM=.
8761
8762                                  ;*ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
8763                                  ;
8764                                  ;*CALL
8765                                  ;*      JSR     PC,CLRQUE
8766
8767  036552  004037  037014          CLRQUE: JSR     R0,SAVR15       ;SAVE R1-R5
8768  036556  012702  036260                  MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
8769  036562  005022                          CLR     (R2)+           ;DRIVES 0 & 1
8770  036564  005022                          CLR     (R2)+           ;DRIVES 2 & 3
8771  036566  005022                          CLR     (R2)+           ;DRIVES 4 & 5
8772  036570  005022                          CLR     (R2)+           ;DRIVES 6 & 7
8773  036572  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING
8774  036576  012701  036330                  MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
8775  036602  012122                  1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
8776  036604  005303                          DEC     R3
8777  036606  001375                          BNE     1$
8778  036610  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
8779  036614  012701  036330                  MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
8780  036620  012122                  2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
8781  036622  005303                          DEC     R3
8782  036624  001375                          BNE     2$
8783  036626  004037  037034                  JSR     R0,GETR15       ;RESTORE R1-R5
8784  036632  000207                          RTS     PC
8785
8786                                  ;*EMPTY THE QUEUE SPECIFIED BY R1
8787                                  ;
8788                                  ;*CALL
8789                                  ;*      MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
8790                                  ;*      JSR     PC,EMPTYQ
8791
8792  036634  105061  036260          EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
8793  036640  006301                          ASL     R1
8794  036642  016161  036270  036310          MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
8795  036650  006201                          ASR     R1
8796  036652  000207                          RTS     PC
8797
8798                                  ;*ROUTINE TO PUT A REQUEST IN QUEUE
8799                                  ;
8800                                  ;*CALL
```

```
 8801                                   ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER
 8802                                   ;*      MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
 8803                                   ;*      JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
 8804                                   ;*      RETURN1                 ;RETURN HERE IF QUEUE IS FULL
 8805                                   ;*      RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
 8806
 8807  036654  122761  000010  036260  DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
 8808  036662  001421                          BEQ     2$              ;BR IF YES-TAKE RETURN1
 8809  036664  105261  036260                  INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
 8810  036670  006301                          ASL     R1
 8811  036672  010271  036270                  MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
 8812  036676  062761  000002  036270          ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
 8813  036704  026161  036270  036332          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
 8814  036712  001003                          BNE     1$              ;BRANCH IF NO
 8815  036714  016161  036330  036270          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
 8816  036722  006201                  1$:     ASR     R1
 8817  036724  005720                          TST     (R0)+           ;TAKE RETURN 2
 8818  036726  000200                  2$:     RTS     R0              ;RETURN TO USER
 8819
 8820                                   ;*ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
 8821                                   ;
 8822                                   ;*CALL
 8823                                   ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 8824                                   ;*      JSR     PC,GETREQ       ;GO GET THE REQUEST
 8825                                   ;*      RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
 8826                                   ;*                              ;R2=0 IF NO REQUEST IN QUEUE
 8827
 8828  036730  005002                  GETREQ: CLR     R2
 8829  036732  105761  036260                  TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
 8830  036736  001404                          BEQ     2$              ;NO---BRANCH
 8831  036740  006301                  1$:     ASL     R1
 8832  036742  017102  036310                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
 8833  036746  006201                          ASR     R1
 8834  036750  000207                  2$:     RTS     PC              ;RETURN TO USER
 8835
 8836                                   ;*ROUTINE TO "POP" THE REQUEST FROM QUEUE
 8837                                   ;
 8838                                   ;*CALL
 8839                                   ;*      MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 8840                                   ;*      JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
 8841                                   ;*      RETURN                  ;R2=ADDRESS OF DPB REMOVED
 8842
 8843  036752  105361  036260          POPQUE: DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
 8844  036756  006301                          ASL     R1
 8845  036760  017102  036310                  MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
 8846  036764  062761  000002  036310          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
 8847  036772  026161  036310  036332          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
 8848  037000  001003                          BNE     1$              ;NO--BRANCH TO EXIT
 8849  037002  016161  036330  036310          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
 8850  037010  006201                  1$:     ASR     R1
 8851  037012  000207                          RTS     PC              ;RETURN TO USER
 8852
 8853                                   ;*ROUTINES TO SAVE R0-R5 AND R1-R5
 8854                                   ;*
```

```
8855                                    ;*CALL: SAVR05
8856                                    ;*      JSR     R0,SAVR05
8857                                    ;*      RETURN                  ;R0-R5 IS ON THE STACK
8858                                    ;*
8859                                    ;*CALL: SAVR15
8860                                    ;*      JSR     R0,SAVR15
8861                                    ;*      RETURN                  ;R1-R5 IS ON THE STACK
8862                                    ;*
8863                                    ;*UPON RETURN FROM SAVR05 AND SAVR15 THE STACK WILL LOOK LIKE:
8864                                    ;*
8865                                    ;*+12   R0
8866                                    ;*+10   R1
8867                                    ;*+06   R2
8868                                    ;*+04   R3
8869                                    ;*+02   R4
8870                                    ;*TOP   R5
8871
8872  037014                            SAVR05:                         ;SAVE R0 BY THE JSR INST.
8873  037014  010146                    SAVR15: MOV     R1,-(SP)        ;SAVE R1
8874  037016  010246                            MOV     R2,-(SP)        ;SAVE R2
8875  037020  010346                            MOV     R3,-(SP)        ;SAVE R3
8876  037022  010446                            MOV     R4,-(SP)        ;SAVE R4
8877  037024  010546                            MOV     R5,-(SP)        ;SAVE R5
8878  037026  016646  000012                    MOV     12(SP),-(SP)    ;GET R0
8879  037032  000200                            RTS     R0
8880
8881                                    ;*ROUTINES TO RESTORE R0-R5 AND R1-R5
8882                                    ;*
8883                                    ;*CALL: GETR05
8884                                    ;*      JSR     R0,GETR05
8885                                    ;*      RETURN                  ;R0-R5 HAVE BEEN RESTORED
8886                                    ;*
8887                                    ;*CALL: GETR15
8888                                    ;*      JSR     R0,GETR15
8889                                    ;*      RETURN                  ;R1-R5 HAVE BEEN RESTORED
8890                                    ;*
8891  037034  011666  000014            GETR15: MOV     (SP),14(SP)     ;POSITION R0
8892  037040  005726                    GETR05: TST     (SP)+           ;POP R0 OF THE JSR OFF OF THE STACK
8893  037042  012605                            MOV     (SP)+,R5        ;RESTORE R5
8894  037044  012604                            MOV     (SP)+,R4        ;RESTORE R4
8895  037046  012603                            MOV     (SP)+,R3        ;RESTORE R3
8896  037050  012602                            MOV     (SP)+,R2        ;RESTORE R2
8897  037052  012601                            MOV     (SP)+,R1        ;RESTORE R1
8898  037054  000200                            RTS     R0              ;RESTORE R0
8899
8900                                    ;******************************************************************
8901
8902                                    .SBTTL  DATA, CONTROL, & STATUS BLOCKS
8903
8904                                    ;******************************************************************
8905
8906                                    ;BLOCK LOCATION EQUATE STATEMENTS
8907
8908          000001                    $FMT    =       1               ;FMT,HCI,ECI OR OFFSET CODE
```

```
8909    000002    $COMND  =    $FMT+1      ;OPERATION CODE
8910    000003    $PSEL   =    $COMND+1    ;PORT SELECT & BITS A16, A17
8911    000004    $WRDM   =    $PSEL+1     ;WORD COUNT (2'S COMP)
8912    000006    $BUF    =    $WRDM+2     ;BUFFER ADDR OR REGISTER TABLE POINTER
8913    000010    $SEC    =    $BUF+2      ;SECTOR ADDRESS OR 1ST REG ADDR
8914    000011    $TRK    =    $SEC+1      ;TRACK ADDRESS OF LAST REG ADDR
8915    000012    $CYL    =    $TRK+1      ;CYLINDER ADDR
8916    000014    $REG    =    $CYL+2      ;REGISTER STORAGE (IF ERROR)
8917    000016    $STATUS =    $REG+2      ;STATUS WORD (SET BY DRIVER)
8918    000020    $WRDL   =    $STATUS+2   ;WORD COUNT (NOT 2'S COMP)
8919    000022    $RETRY  =    $WRDL+2     ;LOWER BYTE = RETRY COUNT, UPPER BYTE = RETRY LIMIT
8920    000024    $MASK   =    $RETRY+2    ;ERROR CHECK MASK
8921    000026    $SSEC   =    $MASK+2     ;SECTOR SIZE FOR CURRENT OPERATION
8922    000030    $CODE   =    $SSEC+2     ;PRESENT COMMAND SELECTION CODE
8923    000031    $PACK   =    $CODE+1     ;WRITE DATA PACK INDICATOR
8924    000032    $PREVO  =    $PACK+1     ;PREVIOUS COMMAND SELECTION CODE
8925    000034    $PATTC  =    $PREVC+2    ;PATTERN CODE
8926    000036    $PREVA  =    $PATTC+2    ;PREVIOUS ADDRESS - TRACK, SECTOR, CYLINDER
8927    000042    $OPERC  =    $PREVA+4    ;OPERATION COUNT
8928    000046    $POSIT  =    $OPERC+4    ;SEEK COUNT
8929    000052    $TRANS  =    $POSIT+4    ;TOTAL BITS XFERED COUNT (R & W)
8930    000056    $READ   =    $TRANS+4    ;TOTAL BITS READ COUNT
8931    000062    $TOTAL  =    $READ+4     ;TOTAL ERRORS (ALL TYPES) COUNT
8932    000064    $SOFT   =    $TOTAL+2    ;'SOFT' ERROR COUNT
8933    000066    $HARD   =    $SOFT+2     ;'HARD' ERROR COUNT
8934    000070    $SKI    =    $HARD+2     ;'SKI' OR 'OCYL' ERROR COUNT
8935    000072    $MISPO  =    $SKI+2      ;PROG DETECTED MISPOSITIONING ERROR S COUNT
8936    000074    $PASSC  =    $MISPO+2    ;PASS COUNTER
8937    000076    $RSAV   =    $PASSC+2    ;STORE THE RANDOM NUMBERS HERE
8938    000102    $NCODE  =    $RSAV+4     ;NEXT OPERATION
8939    000103    $NPATC  =    $NCODE+1    ;NEXT PATTERN
8940    000104    $NSEC   =    $NPATC+1    ;NEXT SECTOR
8941    000105    $NTRK   =    $NSEC+1     ;NEXT TRACK
8942    000106    $NCYL   =    $NTRK+1     ;NEXT CYLINDER
8943    000110    $NWRDL  =    $NCYL+2     ;NEXT BUFFER SIZE
8944    000112    $NEXT   =    $NWRDL+2    ;PARAMETER SELECTION INDICATOR
8945    000114    $BDSEC  =    $NEXT+2     ;BAD SECTOR STORAGE TABLE
8946    000154    $DRVID  =    $BDSEC+40   ;RP04 REGISTER STORAGE
8947    000164    $RHCS1  =    $DRVID+10   ;RP04 REGISTER STORAGE
8948    000166    $RHWC   =    $RHCS1+2
8949    000170    $RHBA   =    $RHWC+2
8950    000172    $RHDA   =    $RHBA+2
8951    000174    $RHCS2  =    $RHCA+2
8952    000176    $RHDS1  =    $RHCS2+2
8953    000200    $RHER1  =    $RHDS1+2
8954    000202    $RHAS   =    $RHER1+2
8955    000204    $RHLA   =    $RHAS+2
8956    000206    $RHDB   =    $RHLA+2
8957    000210    $RHMR   =    $RHDB+2
8958    000212    $RHDT   =    $RHMR+2
8959    000214    $RHSN   =    $RHDT+2
8960    000216    $RHOF   =    $RHSN+2
8961    000220    $RHCA   =    $RHOF+2
8962    000222    $RHCC   =    $RHCA+2
```

```
 8963            000224                  $RHER2  =       $RHCC+2
 8964            000226                  $RHER3  =       $RHER2+2
 8965            000230                  $RHEC1  =       $RHER3+2
 8966            000232                  $RHEC2  =       $RHEC1+2
 8967
 8980
  (1)                                    ;BLOCK FOR DRIVE 0
  (1)
  (1)  037056     000     000            BLK0:   .BYTE   0,0             ;DRIVE NUMBER
  (2)  037060  000000                             .WORD   0
  (2)  037062  000000                             .WORD   0
  (2)  037064  000000                             .WORD   0
  (2)  037066  000000                             .WORD   0
  (2)  037070  000000                             .WORD   0
  (1)  037072  037242                             .WORD   .+150
  (2)  037074  000000                             .WORD   0
  (2)  037076  000000                             .WORD   0
  (2)  037100  000000                             .WORD   0
  (2)  037102  000000                             .WORD   0
  (2)  037104  000000                             .WORD   0
  (2)  037106  000000                             .WORD   0
  (2)  037110  000000                             .WORD   0
  (2)  037112  000000                             .WORD   0
  (2)  037114  000000                             .WORD   0
  (2)  037116  000000                             .WORD   0
  (2)  037120  000000                             .WORD   0
  (2)  037122  000000                             .WORD   0
  (2)  037124  000000                             .WORD   0
  (2)  037126  000000                             .WORD   0
  (2)  037130  000000                             .WORD   0
  (2)  037132  000000                             .WORD   0
  (2)  037134  000000                             .WORD   0
  (2)  037136  000000                             .WORD   0
  (2)  037140  000000                             .WORD   0
  (2)  037142  000000                             .WORD   0
  (2)  037144  000000                             .WORD   0
  (2)  037146  000000                             .WORD   0
  (2)  037150  000000                             .WORD   0
  (2)  037152  000000                             .WORD   0
  (2)  037154  000000                             .WORD   0
  (2)  037156  000000                             .WORD   0
  (2)  037160  000000                             .WORD   0
  (2)  037162  000000                             .WORD   0
  (2)  037164  000000                             .WORD   0
  (2)  037166  000000                             .WORD   0
  (2)  037170  000000                             .WORD   0
  (2)  037172  000000                             .WORD   0
  (2)  037174  000000                             .WORD   0
  (2)  037176  000000                             .WORD   0
  (2)  037200  000000                             .WORD   0
  (2)  037202  000000                             .WORD   0
  (2)  037204  000000                             .WORD   0
  (2)  037206  000000                             .WORD   0
  (2)  037210  000000                             .WORD   0
```

```
(2)  037212  000000                          .WORD   0
(2)  037214  000000                          .WORD   0
(2)  037216  000000                          .WORD   0
(2)  037220  000000                          .WORD   0
(2)  037222  000000                          .WORD   0
(2)  037224  000000                          .WORD   0
(2)  037226  000000                          .WORD   0
(2)  037230  000000                          .WORD   0
(2)  037232  000000                          .WORD   0
(2)  037234  000000                          .WORD   0
(2)  037236  000000                          .WORD   0
(2)  037240  000000                          .WORD   0
(2)  037242  000000                          .WORD   0
(2)  037244  000000                          .WORD   0
(2)  037246  000000                          .WORD   0
(2)  037250  000000                          .WORD   0
(2)  037252  000000                          .WORD   0
(2)  037254  000000                          .WORD   0
(2)  037256  000000                          .WORD   0
(2)  037260  000000                          .WORD   0
(2)  037262  000000                          .WORD   0
(2)  037264  000000                          .WORD   0
(2)  037266  000000                          .WORD   0
(2)  037270  000000                          .WORD   0
(2)  037272  000000                          .WORD   0
(2)  037274  000000                          .WORD   0
(2)  037276  000000                          .WORD   0
(2)  037300  000000                          .WORD   0
(2)  037302  000000                          .WORD   0
(2)  037304  000000                          .WORD   0
(2)  037306  000000                          .WORD   0
(2)  037310  000000                          .WORD   0
(2)  037312  000000                          .WORD   0
(1)
(1)                                 ;BLOCK FOR DRIVE 1
(1)
(1)  037314     001     000         BLK1:    .BYTE   1,0             ;DRIVE NUMBER
(2)  037316  000000                          .WORD   0
(2)  037320  000000                          .WORD   0
(2)  037322  000000                          .WORD   0
(2)  037324  000000                          .WORD   0
(2)  037326  000000                          .WORD   0
(1)  037330  037500                          .WORD   .+150
(2)  037332  000000                          .WORD   0
(2)  037334  000000                          .WORD   0
(2)  037336  000000                          .WORD   0
(2)  037340  000000                          .WORD   0
(2)  037342  000000                          .WORD   0
(2)  037344  000000                          .WORD   0
(2)  037346  000000                          .WORD   0
(2)  037350  000000                          .WORD   0
(2)  037352  000000                          .WORD   0
(2)  037354  000000                          .WORD   0
(2)  037356  000000                          .WORD   0
```

```
(2)  037360  000000                    .WORD   0
(2)  037362  000000                    .WORD   0
(2)  037364  000000                    .WORD   0
(2)  037366  000000                    .WORD   0
(2)  037370  000000                    .WORD   0
(2)  037372  000000                    .WORD   0
(2)  037374  000000                    .WORD   0
(2)  037376  000000                    .WORD   0
(2)  037400  000000                    .WORD   0
(2)  037402  000000                    .WORD   0
(2)  037404  000000                    .WORD   0
(2)  037406  000000                    .WORD   0
(2)  037410  000000                    .WORD   0
(2)  037412  000000                    .WORD   0
(2)  037414  000000                    .WORD   0
(2)  037416  000000                    .WORD   0
(2)  037420  000000                    .WORD   0
(2)  037422  000000                    .WORD   0
(2)  037424  000000                    .WORD   0
(2)  037426  000000                    .WORD   0
(2)  037430  000000                    .WORD   0
(2)  037432  000000                    .WORD   0
(2)  037434  000000                    .WORD   0
(2)  037436  000000                    .WORD   0
(2)  037440  000000                    .WORD   0
(2)  037442  000000                    .WORD   0
(2)  037444  000000                    .WORD   0
(2)  037446  000000                    .WORD   0
(2)  037450  000000                    .WORD   0
(2)  037452  000000                    .WORD   0
(2)  037454  000000                    .WORD   0
(2)  037456  000000                    .WORD   0
(2)  037460  000000                    .WORD   0
(2)  037462  000000                    .WORD   0
(2)  037464  000000                    .WORD   0
(2)  037466  000000                    .WORD   0
(2)  037470  000000                    .WORD   0
(2)  037472  000000                    .WORD   0
(2)  037474  000000                    .WORD   0
(2)  037476  000000                    .WORD   0
(2)  037500  000000                    .WORD   0
(2)  037502  000000                    .WORD   0
(2)  037504  000000                    .WORD   0
(2)  037506  000000                    .WORD   0
(2)  037510  000000                    .WORD   0
(2)  037512  000000                    .WORD   0
(2)  037514  000000                    .WORD   0
(2)  037516  000000                    .WORD   0
(2)  037520  000000                    .WORD   0
(2)  037522  000000                    .WORD   0
(2)  037524  000000                    .WORD   0
(2)  037526  000000                    .WORD   0
(2)  037530  000000                    .WORD   0
(2)  037532  000000                    .WORD   0
```

```
(2)  037534  000000                          .WORD   0
(2)  037536  000000                          .WORD   0
(2)  037540  000000                          .WORD   0
(2)  037542  000000                          .WORD   0
(2)  037544  000000                          .WORD   0
(2)  037546  000000                          .WORD   0
(2)  037550  000000                          .WORD   0
(1)
(1)                                 ;BLOCK FOR DRIVE 2
(1)
(1)  037552     002     000         BLK2:    .BYTE   2,0             ;DRIVE NUMBER
(2)  037554  000000                          .WORD   0
(2)  037556  000000                          .WORD   0
(2)  037560  000000                          .WORD   0
(2)  037562  000000                          .WORD   0
(2)  037564  000000                          .WORD   0
(1)  037566  037736                          .WORD   .+150
(2)  037570  000000                          .WORD   0
(2)  037572  000000                          .WORD   0
(2)  037574  000000                          .WORD   0
(2)  037576  000000                          .WORD   0
(2)  037600  000000                          .WORD   0
(2)  037602  000000                          .WORD   0
(2)  037604  000000                          .WORD   0
(2)  037606  000000                          .WORD   0
(2)  037610  000000                          .WORD   0
(2)  037612  000000                          .WORD   0
(2)  037614  000000                          .WORD   0
(2)  037616  000000                          .WORD   0
(2)  037620  000000                          .WORD   0
(2)  037622  000000                          .WORD   0
(2)  037624  000000                          .WORD   0
(2)  037626  000000                          .WORD   0
(2)  037630  000000                          .WORD   0
(2)  037632  000000                          .WORD   0
(2)  037634  000000                          .WORD   0
(2)  037636  000000                          .WORD   0
(2)  037640  000000                          .WORD   0
(2)  037642  000000                          .WORD   0
(2)  037644  000000                          .WORD   0
(2)  037646  000000                          .WORD   0
(2)  037650  000000                          .WORD   0
(2)  037652  000000                          .WORD   0
(2)  037654  000000                          .WORD   0
(2)  037656  000000                          .WORD   0
(2)  037660  000000                          .WORD   0
(2)  037662  000000                          .WORD   0
(2)  037664  000000                          .WORD   0
(2)  037666  000000                          .WORD   0
(2)  037670  000000                          .WORD   0
(2)  037672  000000                          .WORD   0
(2)  037674  000000                          .WORD   0
(2)  037676  000000                          .WORD   0
(2)  037700  000000                          .WORD   0
```

```
(2)  037702  000000                    .WORD   0
(2)  037704  000000                    .WORD   0
(2)  037706  000000                    .WORD   0
(2)  037710  000000                    .WORD   0
(2)  037712  000000                    .WORD   0
(2)  037714  000000                    .WORD   0
(2)  037716  000000                    .WORD   0
(2)  037720  000000                    .WORD   0
(2)  037722  000000                    .WORD   0
(2)  037724  000000                    .WORD   0
(2)  037726  000000                    .WORD   0
(2)  037730  000000                    .WORD   0
(2)  037732  000000                    .WORD   0
(2)  037734  000000                    .WORD   0
(2)  037736  000000                    .WORD   0
(2)  037740  000000                    .WORD   0
(2)  037742  000000                    .WORD   0
(2)  037744  000000                    .WORD   0
(2)  037746  000000                    .WORD   0
(2)  037750  000000                    .WORD   0
(2)  037752  000000                    .WORD   0
(2)  037754  000000                    .WORD   0
(2)  037756  000000                    .WORD   0
(2)  037760  000000                    .WORD   0
(2)  037762  000000                    .WORD   0
(2)  037764  000000                    .WORD   0
(2)  037766  000000                    .WORD   0
(2)  037770  000000                    .WORD   0
(2)  037772  000000                    .WORD   0
(2)  037774  000000                    .WORD   0
(2)  037776  000000                    .WORD   0
(2)  040000  000000                    .WORD   0
(2)  040002  000000                    .WORD   0
(2)  040004  000000                    .WORD   0
(2)  040006  000000                    .WORD   0
(1)
(1)                             ;BLOCK FOR DRIVE 3
(1)
(1)  040010     003     000     BLK3:   .BYTE   3,0             ;DRIVE NUMBER
(2)  040012  000000                     .WORD   0
(2)  040014  000000                     .WORD   0
(2)  040016  000000                     .WORD   0
(2)  040020  000000                     .WORD   0
(2)  040022  000000                     .WORD   0
(1)  040024  040174                     .WORD   .+150
(2)  040026  000000                     .WORD   0
(2)  040030  000000                     .WORD   0
(2)  040032  000000                     .WORD   0
(2)  040034  000000                     .WORD   0
(2)  040036  000000                     .WORD   0
(2)  040040  000000                     .WORD   0
(2)  040042  000000                     .WORD   0
(2)  040044  000000                     .WORD   0
(2)  040046  000000                     .WORD   0
```

```
(2)  040050  000000                    .WORD   0
(2)  040052  000000                    .WORD   0
(2)  040054  000000                    .WORD   0
(2)  040056  000000                    .WORD   0
(2)  040060  000000                    .WORD   0
(2)  040062  000000                    .WORD   0
(2)  040064  000000                    .WORD   0
(2)  040066  000000                    .WORD   0
(2)  040070  000000                    .WORD   0
(2)  040072  000000                    .WORD   0
(2)  040074  000000                    .WORD   0
(2)  040076  000000                    .WORD   0
(2)  040100  000000                    .WORD   0
(2)  040102  000000                    .WORD   0
(2)  040104  000000                    .WORD   0
(2)  040106  000000                    .WORD   0
(2)  040110  000000                    .WORD   0
(2)  040112  000000                    .WORD   0
(2)  040114  000000                    .WORD   0
(2)  040116  000000                    .WORD   0
(2)  040120  000000                    .WORD   0
(2)  040122  000000                    .WORD   0
(2)  040124  000000                    .WORD   0
(2)  040126  000000                    .WORD   0
(2)  040130  000000                    .WORD   0
(2)  040132  000000                    .WORD   0
(2)  040134  000000                    .WORD   0
(2)  040136  000000                    .WORD   0
(2)  040140  000000                    .WORD   0
(2)  040142  000000                    .WORD   0
(2)  040144  000000                    .WORD   0
(2)  040146  000000                    .WORD   0
(2)  040150  000000                    .WORD   0
(2)  040152  000000                    .WORD   0
(2)  040154  000000                    .WORD   0
(2)  040156  000000                    .WORD   0
(2)  040160  000000                    .WORD   0
(2)  040162  000000                    .WORD   0
(2)  040164  000000                    .WORD   0
(2)  040166  000000                    .WORD   0
(2)  040170  000000                    .WORD   0
(2)  040172  000000                    .WORD   0
(2)  040174  000000                    .WORD   0
(2)  040176  000000                    .WORD   0
(2)  040200  000000                    .WORD   0
(2)  040202  000000                    .WORD   0
(2)  040204  000000                    .WORD   0
(2)  040206  000000                    .WORD   0
(2)  040210  000000                    .WORD   0
(2)  040212  000000                    .WORD   0
(2)  040214  000000                    .WORD   0
(2)  040216  000000                    .WORD   0
(2)  040220  000000                    .WORD   0
(2)  040222  000000                    .WORD   0
```

```
(2)  040224  000000                        .WORD   0
(2)  040226  000000                        .WORD   0
(2)  040230  000000                        .WORD   0
(2)  040232  000000                        .WORD   0
(2)  040234  000000                        .WORD   0
(2)  040236  000000                        .WORD   0
(2)  040240  000000                        .WORD   0
(2)  040242  000000                        .WORD   0
(2)  040244  000000                        .WORD   0
(1)
(1)                                 ;BLOCK FOR DRIVE 4
(1)
(1)  040246     004     000        BLK4:   .BYTE   4,0             ;DRIVE NUMBER
(2)  040250  000000                        .WORD   0
(2)  040252  000000                        .WORD   0
(2)  040254  000000                        .WORD   0
(2)  040256  000000                        .WORD   0
(2)  040260  000000                        .WORD   0
(1)  040262  040432                        .WORD   .+150
(2)  040264  000000                        .WORD   0
(2)  040266  000000                        .WORD   0
(2)  040270  000000                        .WORD   0
(2)  040272  000000                        .WORD   0
(2)  040274  000000                        .WORD   0
(2)  040276  000000                        .WORD   0
(2)  040300  000000                        .WORD   0
(2)  040302  000000                        .WORD   0
(2)  040304  000000                        .WORD   0
(2)  040306  000000                        .WORD   0
(2)  040310  000000                        .WORD   0
(2)  040312  000000                        .WORD   0
(2)  040314  000000                        .WORD   0
(2)  040316  000000                        .WORD   0
(2)  040320  000000                        .WORD   0
(2)  040322  000000                        .WORD   0
(2)  040324  000000                        .WORD   0
(2)  040326  000000                        .WORD   0
(2)  040330  000000                        .WORD   0
(2)  040332  000000                        .WORD   0
(2)  040334  000000                        .WORD   0
(2)  040336  000000                        .WORD   0
(2)  040340  000000                        .WORD   0
(2)  040342  000000                        .WORD   0
(2)  040344  000000                        .WORD   0
(2)  040346  000000                        .WORD   0
(2)  040350  000000                        .WORD   0
(2)  040352  000000                        .WORD   0
(2)  040354  000000                        .WORD   0
(2)  040356  000000                        .WORD   0
(2)  040360  000000                        .WORD   0
(2)  040362  000000                        .WORD   0
(2)  040364  000000                        .WORD   0
(2)  040366  000000                        .WORD   0
(2)  040370  000000                        .WORD   0
```

```
(2)  040372  000000                        .WORD   0
(2)  040374  000000                        .WORD   0
(2)  040376  000000                        .WORD   0
(2)  040400  000000                        .WORD   0
(2)  040402  000000                        .WORD   0
(2)  040404  000000                        .WORD   0
(2)  040406  000000                        .WORD   0
(2)  040410  000000                        .WORD   0
(2)  040412  000000                        .WORD   0
(2)  040414  000000                        .WORD   0
(2)  040416  000000                        .WORD   0
(2)  040420  000000                        .WORD   0
(2)  040422  000000                        .WORD   0
(2)  040424  000000                        .WORD   0
(2)  040426  000000                        .WORD   0
(2)  040430  000000                        .WORD   0
(2)  040432  000000                        .WORD   0
(2)  040434  000000                        .WORD   0
(2)  040436  000000                        .WORD   0
(2)  040440  000000                        .WORD   0
(2)  040442  000000                        .WORD   0
(2)  040444  000000                        .WORD   0
(2)  040446  000000                        .WORD   0
(2)  040450  000000                        .WORD   0
(2)  040452  000000                        .WORD   0
(2)  040454  000000                        .WORD   0
(2)  040456  000000                        .WORD   0
(2)  040460  000000                        .WORD   0
(2)  040462  000000                        .WORD   0
(2)  040464  000000                        .WORD   0
(2)  040466  000000                        .WORD   0
(2)  040470  000000                        .WORD   0
(2)  040472  000000                        .WORD   0
(2)  040474  000000                        .WORD   0
(2)  040476  000000                        .WORD   0
(2)  040500  000000                        .WORD   0
(2)  040502  000000                        .WORD   0
(1)
(1)                                 ;BLOCK FOR DRIVE 5
(1)
(1)  040504     005     000        BLK5:   .BYTE   5,0             ;DRIVE NUMBER
(2)  040506  000000                        .WORD   0
(2)  040510  000000                        .WORD   0
(2)  040512  000000                        .WORD   0
(2)  040514  000000                        .WORD   0
(2)  040516  000000                        .WORD   0
(1)  040520  040670                        .WORD   .+150
(2)  040522  000000                        .WORD   0
(2)  040524  000000                        .WORD   0
(2)  040526  000000                        .WORD   0
(2)  040530  000000                        .WORD   0
(2)  040532  000000                        .WORD   0
(2)  040534  000000                        .WORD   0
(2)  040536  000000                        .WORD   0
```

```
(2)  040540  000000                    .WORD   0
(2)  040542  000000                    .WORD   0
(2)  040544  000000                    .WORD   0
(2)  040546  000000                    .WORD   0
(2)  040550  000000                    .WORD   0
(2)  040552  000000                    .WORD   0
(2)  040554  000000                    .WORD   0
(2)  040556  000000                    .WORD   0
(2)  040560  000000                    .WORD   0
(2)  040562  000000                    .WORD   0
(2)  040564  000000                    .WORD   0
(2)  040566  000000                    .WORD   0
(2)  040570  000000                    .WORD   0
(2)  040572  000000                    .WORD   0
(2)  040574  000000                    .WORD   0
(2)  040576  000000                    .WORD   0
(2)  040600  000000                    .WORD   0
(2)  040602  000000                    .WORD   0
(2)  040604  000000                    .WORD   0
(2)  040606  000000                    .WORD   0
(2)  040610  000000                    .WORD   0
(2)  040612  000000                    .WORD   0
(2)  040614  000000                    .WORD   0
(2)  040616  000000                    .WORD   0
(2)  040620  000000                    .WORD   0
(2)  040622  000000                    .WORD   0
(2)  040624  000000                    .WORD   0
(2)  040626  000000                    .WORD   0
(2)  040630  000000                    .WORD   0
(2)  040632  000000                    .WORD   0
(2)  040634  000000                    .WORD   0
(2)  040636  000000                    .WORD   0
(2)  040640  000000                    .WORD   0
(2)  040642  000000                    .WORD   0
(2)  040644  000000                    .WORD   0
(2)  040646  000000                    .WORD   0
(2)  040650  000000                    .WORD   0
(2)  040652  000000                    .WORD   0
(2)  040654  000000                    .WORD   0
(2)  040656  000000                    .WORD   0
(2)  040660  000000                    .WORD   0
(2)  040662  000000                    .WORD   0
(2)  040664  000000                    .WORD   0
(2)  040666  000000                    .WORD   0
(2)  040670  000000                    .WORD   0
(2)  040672  000000                    .WORD   0
(2)  040674  000000                    .WORD   0
(2)  040676  000000                    .WORD   0
(2)  040700  000000                    .WORD   0
(2)  040702  000000                    .WORD   0
(2)  040704  000000                    .WORD   0
(2)  040706  000000                    .WORD   0
(2)  040710  000000                    .WORD   0
(2)  040712  000000                    .WORD   0
```

```
(2)  040714  000000                        .WORD   0
(2)  040716  000000                        .WORD   0
(2)  040720  000000                        .WORD   0
(2)  040722  000000                        .WORD   0
(2)  040724  000000                        .WORD   0
(2)  040726  000000                        .WORD   0
(2)  040730  000000                        .WORD   0
(2)  040732  000000                        .WORD   0
(2)  040734  000000                        .WORD   0
(2)  040736  000000                        .WORD   0
(2)  040740  000000                        .WORD   0
(1)
(1)                                 ;BLOCK FOR DRIVE 6
(1)
(1)  040742     006     000         BLK6:   .BYTE   6,0             ;DRIVE NUMBER
(2)  040744  000000                        .WORD   0
(2)  040746  000000                        .WORD   0
(2)  040750  000000                        .WORD   0
(2)  040752  000000                        .WORD   0
(2)  040754  000000                        .WORD   0
(1)  040756  041126                        .WORD   .+150
(2)  040760  000000                        .WORD   0
(2)  040762  000000                        .WORD   0
(2)  040764  000000                        .WORD   0
(2)  040766  000000                        .WORD   0
(2)  040770  000000                        .WORD   0
(2)  040772  000000                        .WORD   0
(2)  040774  000000                        .WORD   0
(2)  040776  000000                        .WORD   0
(2)  041000  000000                        .WORD   0
(2)  041002  000000                        .WORD   0
(2)  041004  000000                        .WORD   0
(2)  041006  000000                        .WORD   0
(2)  041010  000000                        .WORD   0
(2)  041012  000000                        .WORD   0
(2)  041014  000000                        .WORD   0
(2)  041016  000000                        .WORD   0
(2)  041020  000000                        .WORD   0
(2)  041022  000000                        .WORD   0
(2)  041024  000000                        .WORD   0
(2)  041026  000000                        .WORD   0
(2)  041030  000000                        .WORD   0
(2)  041032  000000                        .WORD   0
(2)  041034  000000                        .WORD   0
(2)  041036  000000                        .WORD   0
(2)  041040  000000                        .WORD   0
(2)  041042  000000                        .WORD   0
(2)  041044  000000                        .WORD   0
(2)  041046  000000                        .WORD   0
(2)  041050  000000                        .WORD   0
(2)  041052  000000                        .WORD   0
(2)  041054  000000                        .WORD   0
(2)  041056  000000                        .WORD   0
(2)  041060  000000                        .WORD   0
```

```
(2)  041062  000000                          .WORD   0
(2)  041064  000000                          .WORD   0
(2)  041066  000000                          .WORD   0
(2)  041070  000000                          .WORD   0
(2)  041072  000000                          .WORD   0
(2)  041074  000000                          .WORD   0
(2)  041076  000000                          .WORD   0
(2)  041100  000000                          .WORD   0
(2)  041102  000000                          .WORD   0
(2)  041104  000000                          .WORD   0
(2)  041106  000000                          .WORD   0
(2)  041110  000000                          .WORD   0
(2)  041112  000000                          .WORD   0
(2)  041114  000000                          .WORD   0
(2)  041116  000000                          .WORD   0
(2)  041120  000000                          .WORD   0
(2)  041122  000000                          .WORD   0
(2)  041124  000000                          .WORD   0
(2)  041126  000000                          .WORD   0
(2)  041130  000000                          .WORD   0
(2)  041132  000000                          .WORD   0
(2)  041134  000000                          .WORD   0
(2)  041136  000000                          .WORD   0
(2)  041140  000000                          .WORD   0
(2)  041142  000000                          .WORD   0
(2)  041144  000000                          .WORD   0
(2)  041146  000000                          .WORD   0
(2)  041150  000000                          .WORD   0
(2)  041152  000000                          .WORD   0
(2)  041154  000000                          .WORD   0
(2)  041156  000000                          .WORD   0
(2)  041160  000000                          .WORD   0
(2)  041162  000000                          .WORD   0
(2)  041164  000000                          .WORD   0
(2)  041166  000000                          .WORD   0
(2)  041170  000000                          .WORD   0
(2)  041172  000000                          .WORD   0
(2)  041174  000000                          .WORD   0
(2)  041176  000000                          .WORD   0
(1)
(1)                                 ;BLOCK FOR DRIVE 7
(1)
(1)  041200     007     000         BLK7:    .BYTE   7,0             ;DRIVE NUMBER
(2)  041202  000000                          .WORD   0
(2)  041204  000000                          .WORD   0
(2)  041206  000000                          .WORD   0
(2)  041210  000000                          .WORD   0
(2)  041212  000000                          .WORD   0
(1)  041214  041364                          .WORD   .+150
(2)  041216  000000                          .WORD   0
(2)  041220  000000                          .WORD   0
(2)  041222  000000                          .WORD   0
(2)  041224  000000                          .WORD   0
(2)  041226  000000                          .WORD   0
```

```
(2)  041230  000000                    .WORD   0
(2)  041232  000000                    .WORD   0
(2)  041234  000000                    .WORD   0
(2)  041236  000000                    .WORD   0
(2)  041240  000000                    .WORD   0
(2)  041242  000000                    .WORD   0
(2)  041244  000000                    .WORD   0
(2)  041246  000000                    .WORD   0
(2)  041250  000000                    .WORD   0
(2)  041252  000000                    .WORD   0
(2)  041254  000000                    .WORD   0
(2)  041256  000000                    .WORD   0
(2)  041260  000000                    .WORD   0
(2)  041262  000000                    .WORD   0
(2)  041264  000000                    .WORD   0
(2)  041266  000000                    .WORD   0
(2)  041270  000000                    .WORD   0
(2)  041272  000000                    .WORD   0
(2)  041274  000000                    .WORD   0
(2)  041276  000000                    .WORD   0
(2)  041300  000000                    .WORD   0
(2)  041302  000000                    .WORD   0
(2)  041304  000000                    .WORD   0
(2)  041306  000000                    .WORD   0
(2)  041310  000000                    .WORD   0
(2)  041312  000000                    .WORD   0
(2)  041314  000000                    .WORD   0
(2)  041316  000000                    .WORD   0
(2)  041320  000000                    .WORD   0
(2)  041322  000000                    .WORD   0
(2)  041324  000000                    .WORD   0
(2)  041326  000000                    .WORD   0
(2)  041330  000000                    .WORD   0
(2)  041332  000000                    .WORD   0
(2)  041334  000000                    .WORD   0
(2)  041336  000000                    .WORD   0
(2)  041340  000000                    .WORD   0
(2)  041342  000000                    .WORD   0
(2)  041344  000000                    .WORD   0
(2)  041346  000000                    .WORD   0
(2)  041350  000000                    .WORD   0
(2)  041352  000000                    .WORD   0
(2)  041354  000000                    .WORD   0
(2)  041356  000000                    .WORD   0
(2)  041360  000000                    .WORD   0
(2)  041362  000000                    .WORD   0
(2)  041364  000000                    .WORD   0
(2)  041366  000000                    .WORD   0
(2)  041370  000000                    .WORD   0
(2)  041372  000000                    .WORD   0
(2)  041374  000000                    .WORD   0
(2)  041376  000000                    .WORD   0
(2)  041400  000000                    .WORD   0
(2)  041402  000000                    .WORD   0
```

```
  (2)  041404  000000                          .WORD   0
  (2)  041406  000000                          .WORD   0
  (2)  041410  000000                          .WORD   0
  (2)  041412  000000                          .WORD   0
  (2)  041414  000000                          .WORD   0
  (2)  041416  000000                          .WORD   0
  (2)  041420  000000                          .WORD   0
  (2)  041422  000000                          .WORD   0
  (2)  041424  000000                          .WORD   0
  (2)  041426  000000                          .WORD   0
  (2)  041430  000000                          .WORD   0
  (2)  041432  000000                          .WORD   0
  (2)  041434  000000                          .WORD   0
8981
8982                                   ;GENERAL PURPOSE DPB - USED BY 'READHD', 'RECALT', 'OFFSET',&'RTNCTR'
8983
8984  041436  000000  000000  177774   GENDPB: .WORD   0,0,-4,CYLDER
      041444  053554
8985  041446  000000  000000  041456           .WORD   0,0,GENREG,0
      041454  000000
8986  041456  000024                   GENREG: .BLKW   24              ;REGISTER STORAGE IF ERROR
8987
8988                                   ;*****************************************************************
8989
8990                                   .SBTTL  TABLES, CONSTANTS, AND VARIABLE LOCATIONS
8991
8992                                   ;*****************************************************************
8993
9994  041526  000000  000000  000000   ORDERQ: .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF DRIVES PERFORMING COMMANDS
      041534  000000  000000  000000
      041542  000000  000000  000000
8995
8996  041550     000     000     000   ASNLST: .BYTE   0,0,0,0,0,0,0,0 ;-1 IS ASSIGNED UNIT
      041553     000     000     000
      041556     000     000
8997
8998  041560  000000  000000  000000   DUNIT:  .WORD   0,0,0,0,0,0,0,0,0       ;ADDRESSES OF UNIT TO BE DEASSIGNED
      041566  000000  000000  000000
      041574  000000  000000  000000
8999
9000  041602  000000  000000  000000   NEWUNT: .WORD   0,0,0,0,0,0,0,0,0       ;ADDRESSES OF NEWLY ASSIGNED UNITS
      041610  000000  000000  000000
      041616  000000  000000  000000
9001
9002  041624  000000  000000  000000   AVAIL:  .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF UNITS WAITING FOR PARAMETERS
      041632  000000  000000  000000
      041640  000000  000000  000000
9003
9004  041646  000000  000000  000000   WAIT:   .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF UNITS WAITING FOR BUFFERS
      041654  000000  000000  000000
      041662  000000  000000  000000
9005  041670  000000  000000  000000   PARQ:   .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF UNITS WAITING FOR PARAMETERS
      041676  000000  000000  000000
      041704  000000  000000  000000
```

```
 9006
 9007  041712  000000                  BUFTBL: .WORD   0               ;BUFFER ALLOCATION TABLE ENTRY COUNT
 9010  041714  000000  000000                  .WORD   0,0
  (1)  041720  000000  000000                  .WORD   0,0
  (1)  041724  000000  000000                  .WORD   0,0
  (1)  041730  000000  000000                  .WORD   0,0
  (1)  041734  000000  000000                  .WORD   0,0
  (1)  041740  000000  000000                  .WORD   0,0
  (1)  041744  000000  000000                  .WORD   0,0
  (1)  041750  000000  000000                  .WORD   0,0
  (1)  041754  000000  000000                  .WORD   0,0
  (1)  041760  000000  000000                  .WORD   0,0
  (1)  041764  000000  000000                  .WORD   0,0
  (1)  041770  000000  000000                  .WORD   0,0
  (1)  041774  000000  000000                  .WORD   0,0
  (1)  042000  000000  000000                  .WORD   0,0
  (1)  042004  000000  000000                  .WORD   0,0
  (1)  042010  000000  000000                  .WORD   0,0
  (1)  042014  000000  000000                  .WORD   0,0
  (1)  042020  000000  000000                  .WORD   0,0
  (1)  042024  000000  000000                  .WORD   0,0
  (1)  042030  000000  000000                  .WORD   0,0
 9011
 9012  042034                          BLKADR:                 ;ADDRESS OF EACH UNIT BLOCK START
 9015  042034  037056                          .WORD   BLK0
  (1)  042036  037314                          .WORD   BLK1
  (1)  042040  037552                          .WORD   BLK2
  (1)  042042  040010                          .WORD   BLK3
  (1)  042044  040246                          .WORD   BLK4
  (1)  042046  040504                          .WORD   BLK5
  (1)  042050  040742                          .WORD   BLK6
  (1)  042052  041200                          .WORD   BLK7
 9016
```

```
 9018  042054     151                    COMTBL: .BYTE   WCKD            ;WRITE CHECK DATA
 9019  042055     153                            .BYTE   WCKHD           ;WRITE CHECK HEADER AND DATA
 9020  042056     161                            .BYTE   WRTDAT          ;WRITE DATA
 9021  042057     163                            .BYTE   WRTHD           ;WRITE HEADER AND DATA
 9022  042060     171                            .BYTE   RDDAT           ;READ DATA
 9023  042061     173                            .BYTE   RCRD            ;READ HEADER AND DATA
 9024
 9025  042062  041527  042113  020040  OPMNEM: .ASCIZ  /WCKD   /        ;WRITE CHECK DATA
       042070  000040
 9026  042072  041527  044113  020104          .ASCIZ  /WCKHD  /        ;WRITE CHECK HEADER & DATA
       042100  000040
 9027  042102  051127  042124  052101          .ASCIZ  /WRTDAT /        ;WRITE DATA
       042110  000040
 9028  042112  051127  044124  020104          .ASCIZ  /WRTHD  /        ;WRITE HEADER & DATA
       042120  000040
 9029  042122  042122  040504  020124          .ASCIZ  /RDDAT  /        ;READ DATA
       042130  000040
 9030  042132  042122  042110  020040          .ASCIZ  /RDHD   /        ;READ HEADER & DATA
       042140  000040
 9031
 9032  042142     000                    OFFCOD: .BYTE   0               ;OFFSET CODE TABLE
 9033  042143     020                            .BYTE   20              ;+400 U INCHES
 9034  042144     220                            .BYTE   220             ;-400 U INCHES
 9035  042145     040                            .BYTE   40              ;+800 U INCHES
 9036  042146     240                            .BYTE   240             ;-800 U INCHES
 9037  042147     060                            .BYTE   60              ;+1200 U INCHES
 9038  042150     260     000                    .BYTE   260,0           ;-1200 U INCHES, TERMINATOR
 9039
 9040  042152  042170                    OFMTBL: .WORD   OFMSG0          ;1ST OFFSET MESSAGE
 9041  042154  042223                            .WORD   OFMSG1          ;2ND OFFSET MESSAGE
 9042  042156  042[illegible]7                   .WORD   OFMSG2          ;3RD OFFSET MESSAGE
 9043  042160  04[illegible]3                    .WORD   OFMSG3          ;4TH OFFSET MESSAGE
 9044  042162  04[illegible]7                    .WORD   OFMSG4          ;5TH OFFSET MESSAGE
 9045  042164  04[illegible]03                   .WORD   OFMSG5          ;6TH OFFSET MESSAGE
 9046  042166  042440                          .WORD   OFMSG6          ;7TH OFFSET MESSAGE
 9047
 9048  042170  043101  042524  020122  OFMSG0: .ASCIZ  /AFTER RETRY WITHOUT OFFSET/
       042176  042522  051124  020131
       042204  044527  044124  052517
       042212  020124  043117  051506
       042220  052105     000
 9049  042223     101  020124  043117  OFMSG1: .ASCIZ  /AT OFFSET +400 MICRO-INCHES/
       042230  051506  052105  025440
       042236  030064  020060  044515
       042244  051103  026517  047111
       042252  044103  051505     000
 9050  042257     101  020124  043117  OFMSG2: .ASCIZ  /AT OFFSET -400 MICRO-INCHES/
       042264  051506  052105  026440
       042272  030064  020060  044515
       042300  051103  026517  047111
       042306  044103  051505     000
 9051  042313     101  020124  043117  OFMSG3: .ASCIZ  /AT OFFSET +800 MICRO-INCHES/
       042320  051506  052105  025440
       042326  030070  020060  044515
```

```
                042334  051103  026517  047111
                042342  044103  051505     000
9052            042347     101  020124  043117  OFMSG4: .ASCIZ  /AT OFFSET -800 MICRO-INCHES/
                042354  051506  052105  026440
                042362  030070  020060  044515
                042370  051103  026517  047111
                042376  044103  051505     000
9053            042403     101  020124  043117  OFMSG5: .ASCIZ  /AT OFFSET +1200 MICRO-INCHES/
                042410  051505  052105  025440
                042416  031061  030060  046440
                042424  041511  047522  044455
                042432  041516  042510  000123
9054            042440  052101  047440  043106  OFMSG6: .ASCIZ  /AT OFFSET -1200 MICRO-INCHES/
                042446  042523  020124  030455
                042454  030062  020060  044515
                042462  051103  026517  047111
                042470  044103  051505     000
9055
9056                    042476                          .EVEN
9057
9058                                                    ;ERROR MESSAGE LINE 2 POINTER TABLE
9059
9060            042476  046462                  DT14.T: .WORD   DT14.0          ;DT14 ADDRESSES
9061            042500  046504                          .WORD   DT14.1
9062            042502  046526                          .WORD   DT14.2
9063            042504  046550                          .WORD   DT14.3
9064            042506  046572                          .WORD   DT14.4
9065            042510  046614                          .WORD   DT14.5
9066            042512  046636                          .WORD   DT14.6
9067            042514  046660                          .WORD   DT14.7
9068
9069            042516  046702                  DT15.T: .WORD   DT15.0          ;DT15 ADDRESSES
9070            042520  046724                          .WORD   DT15.1
9071            042522  046746                          .WORD   DT15.2
9072            042524  046770                          .WORD   DT15.3
9073            042526  047012                          .WORD   DT15.4
9074            042530  047034                          .WORD   DT15.5
9075            042532  047056                          .WORD   DT15.6
9076            042534  047100                          .WORD   DT15.7
9077
9078            042536  047122                  DT16.T: .WORD   DT16.0          ;DT16 ADDRESSES
9079            042540  047136                          .WORD   DT16.1
9080            042542  047152                          .WORD   DT16.2
9081            042544  047166                          .WORD   DT16.3
9082            042546  047202                          .WORD   DT16.4
9083            042550  047216                          .WORD   DT16.5
9084            042552  047232                          .WORD   DT16.6
9085            042554  047246                          .WORD   DT16.7
9086
9087                                            ;*****************************************************************
9088
9089                                            .SBTTL  DATA PATTERNS
9090
9091                                            ;*****************************************************************
```

```
 9092
 9093  042556  000000                STNDAT: .WORD   0               ;STANDARD DATA PATTERN POINTER TABLE
 9094  042560  042662                        .WORD   DATA1
 9095  042562  042722                        .WORD   DATA1+40
 9096  042564  042762                        .WORD   DATA1+100
 9097  042566  043022                        .WORD   DATA1+140
 9098  042570  043062                        .WORD   DATA1+200
 9099  042572  043122                        .WORD   DATA1+240
 9100  042574  043162                        .WORD   DATA1+300
 9101  042576  043222                        .WORD   DATA1+340
 9102  042600  043262                        .WORD   DATA1+400
 9103  042602  043322                        .WORD   DATA1+440
 9104  042604  043362                        .WORD   DATA1+500
 9105  042606  043422                        .WORD   DATA1+540
 9106  042610  043462                        .WORD   DATA1+600
 9107  042612  043522                        .WORD   DATA1+640
 9108  042614  043562                        .WORD   DATA1+700
 9109  042616  042622                        .WORD   DATA0           ;ZEROES
 9110  042620  043524                        .WORD   DATA1+642       ;ONES
 9111
 9112  042622  000000                DATA0:  .WORD   0               ;DUMMY DATA PATTERN
 9115  042624  000000                        .WORD   0
  (1)  042626  000000                        .WORD   0
  (1)  042630  000000                        .WORD   0
  (1)  042632  000000                        .WORD   0
  (1)  042634  000000                        .WORD   0
  (1)  042636  000000                        .WORD   0
  (1)  042640  000000                        .WORD   0
  (1)  042642  000000                        .WORD   0
  (1)  042644  000000                        .WORD   0
  (1)  042646  000000                        .WORD   0
  (1)  042650  000000                        .WORD   0
  (1)  042652  000000                        .WORD   0
  (1)  042654  000000                        .WORD   0
  (1)  042656  000000                        .WORD   0
  (1)  042660  000000                        .WORD   0
 9116
 9117  042662  000001                DATA1:  .WORD   000001  ;STANDARD PATTERN 1
 9118  042664  000003                        .WORD   000003
 9119  042666  000007                        .WORD   000007
 9120  042670  000017                        .WORD   000017
 9121  042672  000037                        .WORD   000037
 9122  042674  000077                        .WORD   000077
 9123  042676  000177                        .WORD   000177
 9124  042700  000377                        .WORD   000377
 9125  042702  000777                        .WORD   000777
 9126  042704  001777                        .WORD   001777
 9127  042706  003777                        .WORD   003777
 9128  042710  007777                        .WORD   007777
 9129  042712  017777                        .WORD   017777
 9130  042714  037777                        .WORD   037777
 9131  042716  077777                        .WORD   077777
 9132  042720  177777                        .WORD   177777
 9133
```

```
 9134  042722  177776                    .WORD   177776  ;STANDARD PATTERN 2
 9135  042724  177774                    .WORD   177774
 9136  042726  177770                    .WORD   177770
 9137  042730  177760                    .WORD   177760
 9138  042732  177740                    .WORD   177740
 9139  042734  177700                    .WORD   177700
 9140  042736  177600                    .WORD   177600
 9141  042740  177400                    .WORD   177400
 9142  042742  177000                    .WORD   177000
 9143  042744  176000                    .WORD   176000
 9144  042746  174000                    .WORD   174000
 9145  042750  170000                    .WORD   170000
 9146  042752  160000                    .WORD   160000
 9147  042754  140000                    .WORD   140000
 9148  042756  100000                    .WORD   100000
 9149  042760  000000                    .WORD   000000
 9150
 9151  042762  000000                    .WORD   000000  ;STANDARD PATTERN 3
 9152  042764  000000                    .WORD   000000
 9153  042766  000000                    .WORD   000000
 9154  042770  177777                    .WORD   177777
 9155  042772  177777                    .WORD   177777
 9156  042774  177777                    .WORD   177777
 9157  042776  000000                    .WORD   000000
 9158  043000  000000                    .WORD   000000
 9159  043002  177777                    .WORD   177777
 9160  043004  177777                    .WORD   177777
 9161  043006  000000                    .WORD   000000
 9162  043010  177777                    .WORD   177777
 9163  043012  000000                    .WORD   000000
 9164  043014  177777                    .WORD   177777
 9165  043016  000000                    .WORD   000000
 9166  043020  177777                    .WORD   177777
 9167
 9168  043022  000000                    .WORD   000000  ;STANDARD PATTERN 4
 9169  043024  010421                    .WORD   010421
 9170  043026  021042                    .WORD   021042
 9171  043030  031463                    .WORD   031463
 9172  043032  042104                    .WORD   042104
 9173  043034  052525                    .WORD   052525
 9174  043036  063146                    .WORD   063146
 9175  043040  073567                    .WORD   073567
 9176  043042  104210                    .WORD   104210
 9177  043044  114631                    .WORD   114631
 9178  043046  125252                    .WORD   125252
 9179  043050  135673                    .WORD   135673
 9180  043052  146314                    .WORD   146314
 9181  043054  156735                    .WORD   156735
 9182  043056  167356                    .WORD   167356
 9183  043060  177777                    .WORD   177777
 9184
 9185  043062  052525                    .WORD   052525  ;STANDARD PATTERN 5
 9186  043064  052525                    .WORD   052525
 9187  043066  052525                    .WORD   052525
```

```
 9188  043070  125252                    .WORD   125252
 9189  043072  125252                    .WORD   125252
 9190  043074  125252                    .WORD   125252
 9191  043076  052525                    .WORD   052525
 9192  043100  052525                    .WORD   052525
 9193  043102  125252                    .WORD   125252
 9194  043104  125252                    .WORD   125252
 9195  043106  052525                    .WORD   052525
 9196  043110  125252                    .WORD   125252
 9197  043112  052525                    .WORD   052525
 9198  043114  125252                    .WORD   125252
 9199  043116  052525                    .WORD   052525
 9200  043120  125252                    .WORD   125252
 9201
 9202  043122  007417                    .WORD   007417  ;STANDARD PATTERN 6
 9203  043124  007417                    .WORD   007417
 9204  043126  007417                    .WORD   007417
 9205  043130  170360                    .WORD   170360
 9206  043132  170360                    .WORD   170360
 9207  043134  170360                    .WORD   170360
 9208  043136  007417                    .WORD   007417
 9209  043140  007417                    .WORD   007417
 9210  043142  170360                    .WORD   170360
 9211  043144  170360                    .WORD   170360
 9212  043146  007417                    .WORD   007417
 9213  043150  170360                    .WORD   170360
 9214  043152  007417                    .WORD   007417
 9215  043154  170360                    .WORD   170360
 9216  043156  007417                    .WORD   007417
 9217  043160  170360                    .WORD   170360
 9218
 9219  043162  026455                    .WORD   026455  ;STANDARD PATTERN 7
 9220  043164  026455                    .WORD   026455
 9221  043166  026455                    .WORD   026455
 9222  043170  151322                    .WORD   151322
 9223  043172  151322                    .WORD   151322
 9224  043174  151322                    .WORD   151322
 9225  043176  026455                    .WORD   026455
 9226  043200  026455                    .WORD   026455
 9227  043202  151322                    .WORD   151322
 9228  043204  151322                    .WORD   151322
 9229  043206  026455                    .WORD   026455
 9230  043210  151322                    .WORD   151322
 9231  043212  026455                    .WORD   026455
 9232  043214  151322                    .WORD   151322
 9233  043216  026455                    .WORD   026455
 9234  043220  151322                    .WORD   151322
 9235
 9236  043222  165555                    .WORD   165555  ;STANDARD PATTERN 8
 9237  043224  133333                    .WORD   133333
 9238  043226  165555                    .WORD   165555
 9239  043230  133333                    .WORD   133333
 9240  043232  165555                    .WORD   165555
 9241  043234  133333                    .WORD   133333
```

```
 9242  043236  165555                    .WORD   165555
 9243  043240  133333                    .WORD   133333
 9244  043242  165555                    .WORD   165555
 9245  043244  133333                    .WORD   133333
 9246  043246  165555                    .WORD   165555
 9247  043250  133333                    .WORD   133333
 9248  043252  165555                    .WORD   165555
 9249  043254  133333                    .WORD   133333
 9250  043256  165555                    .WORD   165555
 9251  043260  133333                    .WORD   133333
 9252
 9253  043262  000001                    .WORD   000001  ;STANDARD PATTERN 9
 9254  043264  000002                    .WORD   000002
 9255  043266  000004                    .WORD   000004
 9256  043270  000010                    .WORD   000010
 9257  043272  000020                    .WORD   000020
 9258  043274  000040                    .WORD   000040
 9259  043276  000100                    .WORD   000100
 9260  043300  000200                    .WORD   000200
 9261  043302  000400                    .WORD   000400
 9262  043304  001000                    .WORD   001000
 9263  043306  002000                    .WORD   002000
 9264  043310  004000                    .WORD   004000
 9265  043312  010000                    .WORD   010000
 9266  043314  020000                    .WORD   020000
 9267  043316  040000                    .WORD   040000
 9268  043320  100000                    .WORD   100000
 9269
 9270  043322  177776                    .WORD   177776  ;STANDARD PATTERN 10
 9271  043324  177775                    .WORD   177775
 9272  043326  177773                    .WORD   177773
 9273  043330  177767                    .WORD   177767
 9274  043332  177757                    .WORD   177757
 9275  043334  177737                    .WORD   177737
 9276  043336  177677                    .WORD   177677
 9277  043340  177577                    .WORD   177577
 9278  043342  177377                    .WORD   177377
 9279  043344  176777                    .WORD   176777
 9280  043346  175777                    .WORD   175777
 9281  043350  173777                    .WORD   173777
 9282  043352  167777                    .WORD   167777
 9283  043354  157777                    .WORD   157777
 9284  043356  137777                    .WORD   137777
 9285  043360  077777                    .WORD   077777
 9286
 9287  043362  172666                    .WORD   172666  ;STANDARD PATTERN 11
 9288  043364  155555                    .WORD   155555
 9289  043366  172666                    .WORD   172666
 9290  043370  155555                    .WORD   155555
 9291  043372  172666                    .WORD   172666
 9292  043374  155555                    .WORD   155555
 9293  043376  172666                    .WORD   172666
 9294  043400  155555                    .WORD   155555
 9295  043402  172666                    .WORD   172666
```

```
 9296  043404  155555                    .WORD   155555
 9297  043406  172666                    .WORD   172666
 9298  043410  155555                    .WORD   155555
 9299  043412  172666                    .WORD   172666
 9300  043414  155555                    .WORD   155555
 9301  043416  172666                    .WORD   172666
 9302  043420  155555                    .WORD   155555
 9303
 9304  043422  077777                    .WORD   077777  ;STANDARD PATTERN 12
 9305  043424  137777                    .WORD   137777
 9306  043426  157777                    .WORD   157777
 9307  043430  167777                    .WORD   167777
 9308  043432  173777                    .WORD   173777
 9309  043434  175777                    .WORD   175777
 9310  043436  176777                    .WORD   176777
 9311  043440  177377                    .WORD   177377
 9312  043442  177577                    .WORD   177577
 9313  043444  177677                    .WORD   177677
 9314  043446  177737                    .WORD   177737
 9315  043450  177757                    .WORD   177757
 9316  043452  177767                    .WORD   177767
 9317  043454  177773                    .WORD   177773
 9318  043456  177775                    .WORD   177775
 9319  043460  177776                    .WORD   177776
 9320
 9321  043462  153333                    .WORD   153333  ;STANDARD PATTERN 13
 9322  043464  066667                    .WORD   066667
 9323  043466  153333                    .WORD   153333
 9324  043470  066667                    .WORD   066667
 9325  043472  153333                    .WORD   153333
 9326  043474  066667                    .WORD   066667
 9327  043476  153333                    .WORD   153333
 9328  043500  066667                    .WORD   066667
 9329  043502  153333                    .WORD   153333
 9330  043504  066667                    .WORD   066667
 9331  043506  153333                    .WORD   153333
 9332  043510  066667                    .WORD   066667
 9333  043512  153333                    .WORD   153333
 9334  043514  066667                    .WORD   066667
 9335  043516  153333                    .WORD   153333
 9336  043520  066667                    .WORD   066667
 9337
 9338  043522  000000                    .WORD   000000  ;STANDARD PATTERN 14
 9339  043524  177777                    .WORD   177777
 9340  043526  177777                    .WORD   177777
 9341  043530  177777                    .WORD   177777
 9342  043532  177777                    .WORD   177777
 9343  043534  177777                    .WORD   177777
 9344  043536  177777                    .WORD   177777
 9345  043540  177777                    .WORD   177777
 9346  043542  177777                    .WORD   177777
 9347  043544  177777                    .WORD   177777
 9348  043546  177777                    .WORD   177777
 9349  043550  177777                    .WORD   177777
```

```
 9350  043552  177777                          .WORD   177777
 9351  043554  177777                          .WORD   177777
 9352  043556  177777                          .WORD   177777
 9353  043560  177777                          .WORD   177777
 9354
 9355  043562  177777                          .WORD   177777  ;STANDARD PATTERN 15
 9356  043564  000000                          .WORD   000000
 9357  043566  000000                          .WORD   000000
 9358  043570  000000                          .WORD   000000
 9359  043572  000000                          .WORD   000000
 9360  043574  000000                          .WORD   000000
 9361  043576  000000                          .WORD   000000
 9362  043600  000000                          .WORD   000000
 9363  043602  000000                          .WORD   000000
 9364  043604  000000                          .WORD   000000
 9365  043606  000000                          .WORD   000000
 9366  043610  000000                          .WORD   000000
 9367  043612  000000                          .WORD   000000
 9368  043614  000000                          .WORD   000000
 9369  043616  000000                          .WORD   000000
 9370  043620  000000                          .WORD   000000
 9371
 9372                                  ;*****************************************************************
 9373
 9374                                  .SBTTL  PRINTER/TELETYPE MESSAGES
 9375
 9376                                  ;*****************************************************************
 9377
 9378  043622  046111  042514  040507  EM1:    .ASCIZ  /ILLEGAL RH11 INTERRUPT (SC=0 OR RHAS=0)/
       043630  020114  044122  030461
       043636  044440  052116  051105
       043644  052522  052120  024040
       043652  041523  030075  047440
       043660  020122  044122  051501
       043666  030075  000051
 9379
 9380  043672  047125  054105  042520  EM2:    .ASCIZ  /UNEXPECTED ATTENTION/
       043700  052103  042105  040440
       043706  052124  047105  044524
       043714  047117     000
 9381
 9382  043717     103  047117  051124  EM3:    .ASCIZ  /CONTROL BUS PARITY ERROR DETECTED BY THE RH11/
       043724  046117  041040  051525
       043732  050040  051101  052111
       043740  020131  051105  047522
       043746  020122  042504  042524
       043754  052103  042105  041040
       043762  020131  044124  020105
       043770  044122  030461     000
 9383
 9384  043775     103  047117  051124  EM4:    .ASCIZ  /CONTROL BUS PARITY ERROR DETECTED BY THE RP04/
       044002  046117  041040  051525
       044010  050040  051101  052111
       044016  020131  051105  047522
```

```
        044024  020122  042504  042524
        044032  052103  042105  041040
        044040  020131  044124  020105
        044046  050122  032060     000
 9385
 9386   044053     101  052124  047105  EM5:    .ASCIZ  /ATTENTION FROM AN OFFLINE UNIT/
        044060  044524  047117  043040
        044066  047522  020115  047101
        044074  047440  043106  044514
        044102  042516  052440  044516
        044110  000124
 9387
 9388   044112  047125  047503  051122  EM10:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
        044120  041505  040524  046102
        044126  020105  040515  051523
        044134  052502  020123  040520
        044142  044522  054524  042440
        044150  051122  051117     000
 9389
 9390   044155     106  052101  046101  EM11:   .ASCIZ  /FATAL MASSBUS PARITY ERROR/
        044162  046440  051501  041123
        044170  051525  050040  051101
        044176  052111  020131  051105
        044204  047522  000122
 9391
 9392   044210  042520  051522  051511  EM12:   .ASCIZ  /PERSISTENT DEVICE UNSAFE/
        044216  042524  052116  042040
        044224  053105  041511  020105
        044232  047125  040523  042506
        044240     000
 9393
 9394   044241     117  042520  040522  EM13:   .ASCIZ  /OPERATION NOT COMPLETED WITHIN TIME LIMIT/
        044246  044524  047117  047040
        044254  052117  041440  046517
        044262  046120  052105  042105
        044270  053440  052111  044510
        044276  020116  044524  042515
        044304  046040  046511  052111
        044312     000
 9395
 9396   044313     125  044516  020124  EM14:   .ASCIZ  /UNIT WENT OFFLINE/
        044320  042527  052116  047440
        044326  043106  044514  042516
        044334     000
 9397
 9398   044335     116  020117  042522  EM15:   .ASCIZ  /NO RESPONSE TO PORT REQUEST/
        044342  050123  047117  042523
        044350  052040  020117  047520
        044356  052122  051040  050505
        044364  042525  052123     000
 9399
 9400   044371     110  040505  042504  EM20:   .ASCIZ  /HEADER CRC ERROR/
        044376  020122  051103  020103
        044404  051105  047522  000122
```

```
9401
9402  044412 040504 040524 041440  EM21:   .ASCIZ  /DATA CHECK ('DCK') ERROR/
      044420 042510 045503 024040
      044426 042047 045503 024447
      044434 042440 051122 051117
      044442    000
9403
9404  044443    127 044522 042524  EM22:   .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') SET/
      044450 041440 042510 045503
      044456 042440 051122 051117
      044464 026440 042040 052101
      044472 020101 044103 041505
      044500 020113 023450 041504
      044506 023513 020051 042523
      044514 000124
9405
9406  044516 051127 052111 020105  EM23:   .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') NOT SET/
      044524 044103 041505 020113
      044532 051105 047522 020122
      044540 020055 040504 040524
      044546 041440 042510 045503
      044554 024040 042047 045503
      044562 024447 047040 052117
      044570 051440 052105    000
9407
9408  044575    110 040505 042504  EM24:   .ASCIZ  /HEADER READ ERROR - 'FMT' BIT DROPPED/
      044602 020122 042522 042101
      044610 042440 051122 051117
      044616 026440 023440 046506
      044624 023524 041040 052111
      044632 042040 047522 050120
      044640 042105    000
9409
9410  044643    110 040505 042504  EM25:   .ASCIZ  /HEADER READ ERROR - HEADER COMPARE ('HCE') ERROR/
      044650 020122 042522 042101
      044656 042440 051122 051117
      044664 026440 044040 040505
      044672 042504 020122 047503
      044700 050115 051101 020105
      044706 023450 041510 023505
      044714 020051 051105 047522
      044722 000122
9411
9412  044724 047506 046522 052101  EM26:   .ASCIZ  /FORMAT ERROR ('FER')/
      044732 042440 051122 051117
      044740 024040 043047 051105
      044746 024447    000
9413
9414  044751    110 040505 042504  EM27:   .ASCIZ  /HEADER COMPARE ('HCE') ERROR/
      044756 020122 047503 050115
      044764 051101 020105 023450
      044772 041510 023505 020051
      045000 051105 047522 000122
9415
```

```
9416  045006 044515 041523 046105  EM30:   .ASCIZ  /MISCELLANEOUS DRIVE ERROR/
      045014 040514 042516 052517
      045022 020123 051104 053111
      045030 020105 051105 047522
      045036 000122
9417
9418  045040 050117 051105 052101  EM31:   .ASCIZ  /OPERATION INCOMPLETE ('OPI') ERROR/
      045046 047511 020116 047111
      045054 047503 050115 042514
      045062 042524 024040 047447
      045070 044520 024447 042440
      045076 051122 051117    000
9419
9420  045103    104 044522 042526  EM32:   .ASCIZ  /DRIVE TIMING ('DTE') ERROR/
      045110 052040 046511 047111
      045116 020107 023450 052104
      045124 023505 020051 051105
      045132 047522 000122
9421
9422  045136 040520 044522 054524  EM33:   .ASCIZ  /PARITY ('PAR') ERROR AFTER OPERATION STARTED/
      045144 024040 050047 051101
      045152 024447 042440 051122
      045160 051117 040440 052106
      045166 051105 047440 042520
      045174 040522 044524 047117
      045202 051440 040524 052122
      045210 042105    000
9423
9424  045213    127 044522 042524  EM34:   .ASCIZ  /WRITE CLOCK FAILURE ('WCF') ERROR/
      045220 041440 047514 045503
      045226 043040 044501 052514
      045234 042522 024040 053447
      045242 043103 024447 042440
      045250 051122 051117    000
9425
9426  045255    111 053116 046101  EM35:   .ASCIZ  /INVALID ADDRESS ('IAE') ERROR/
      045262 042111 040440 042104
      045270 042522 051523 024040
      045276 044447 042501 024447
      045304 042440 051122 051117
      045312    000
9427
9428  045313    127 044522 042524  EM36:   .ASCIZ  /WRITE LOCK ('WLE') ERROR/
      045320 046040 041517 020113
      045326 023450 046127 023505
      045334 020051 051105 047522
      045342 000122
9429
9430  045344 044122 030461 047440  EM40:   .ASCIZ  /RH11 OR UNIBUS TRANSFER ERROR/
      045352 020122 047125 041111
      045360 051525 052040 040522
      045366 051516 042506 020122
      045374 051105 047522 000122
9431
```

```
 9432  045402 052502 020123 042101 EM41:   .ASCIZ  /BUS ADDRESS OR WORD COUNT INCORRECT/
       045410 051104 051505 020123
       045416 051117 053440 051117
       045424 020104 047503 047125
       045432 020124 047111 047503
       045440 051122 041505 000124
 9433
 9434  045446 040504 040524 041440 EM42:   .ASCIZ  /DATA COMPARE ERRORS - NO RP04 ERROR DETECTED/
       045454 046517 040520 042522
       045462 042440 051122 051117
       045470 020123 020055 047516
       045476 051040 030120 020064
       045504 051105 047522 020122
       045512 042504 042524 052103
       045520 042105    000
 9435
 9436  045523    103 047101 052047 EM43:   .ASCIZ  /CAN'T MATCH DATA READ WITH A PATTERN/
       045530 046440 052101 044103
       045536 042040 052101 020101
       045544 042522 042101 053440
       045552 052111 020110 020101
       045560 040520 052124 051105
       045566 000116
 9437
 9438  045570 051105 047522 020122 EM44:   .ASCIZ  /ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11/
       045576 044502 024124 024523
       045604 051440 052105 020054
       045612 052502 020124 047516
       045620 042440 051122 051117
       045626 051440 043511 040516
       045634 042514 020104 054502
       045642 052040 042510 051040
       045650 030510 000061
 9439  045654 042523 045505 044440 EM50:   .ASCIZ  /SEEK INCOMPLETE OR OFF CYLINDER ERROR/
       045662 041516 046517 046120
       045670 052105 020105 051117
       045676 047440 043106 041440
       045704 046131 047111 042504
       045712 020122 051105 047522
       045720 000122
 9440
 9441  045722 051120 043517 040522 EM51:   .ASCIZ  /PROGRAM DETECTED POSITIONING ERROR/
       045730 020115 042504 042524
       045736 052103 042105 050040
       045744 051517 052111 047511
       045752 044516 043516 042440
       045760 051122 051117    000
 9442
 9443  045765    104 053105 041511 EM60:   .ASCIZ  /DEVICE UNSAFE/
       045772 020105 047125 040523
       046000 042506    000
 9444
 9445  046003    040 020040 042040 DH2:    .ASCIZ  /    DRV RHDS1   RHER1   RHER2   RHER3   RHAS/
       046010 053122 051040 042110
```

```
                046016  030523  020040  051040
                046024  042510  030522  020040
                046032  051040  042510  031122
                046040  020040  051040  042510
                046046  031522  020040  051040
                046054  040510  000123
 9446
 9447   046060  020040  020040  051104  DH3:    .ASCIZ  /    DRV REG ADR  DATA/
        046066  020126  042522  020107
        046074  042101  020122  042040
        046102  052101  000101
 9448
 9449   046106  020040  020040  051104  DH4:    .ASCIZ  /    DRV REG ADR    GOOD    BAD/
        046114  020126  042522  020107
        046122  042101  020122  020040
        046130  043440  047517  020104
        046136  020040  041040  042101
        046144     000
 9450
 9451   046145     104  053122  051040  DH14:   .ASCII  /DRV RHCS1   RHCS2   RHDS1   RHER1   /
        046152  041510  030523  020040
        046160  051040  041510  031123
        046166  020040  051040  042110
        046174  030523  020040  051040
        046202  042510  030522  020040
        046210     040
 9452   046211     122  042510  031122          .ASCIZ  /RHER2   RHER3   RHEC1   RHEC2/<15><12>
        046216  020040  051040  042510
        046224  031522  020040  051040
        046232  042510  030503  020040
        046240  051040  042510  031103
        046246  005015     000
 9453
 9454   046251     122  053510  020103  DH15:   .ASCII  /RHWC    RHBA    RHDA    RHAS    RHLA    /
        046256  020040  051040  041110
        046264  020101  020040  051040
        046272  042110  020101  020040
        046300  051040  040510  020123
        046306  020040  051040  046110
        046314  020101  020040     040
 9455   046321     122  042110  020102          .ASCIZ  /RHDB    RHMR    RHDT/<15><12>
        046326  020040  051040  046510
        046334  020122  020040  051040
        046342  042110  006524  000012
 9456
 9457   046350  044122  047123  020040  DH16:   .ASCIZ  /RHSN    RHOF    RHCA    RHCC    STATUS/<15><12>
        046356  020040  044122  043117
        046364  020040  020040  044122
        046372  040503  020040  020040
        046400  044122  041503  020040
        046406  020040  052123  052101
        046414  051525  005015     000
 9458
 9459           046422                          .EVEN
```

```
 9460
 9461  046422  001250  030446  030450  DT2:    .WORD   DRIVE,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6,ATTN,0
       046430  030452  030454  001252
       046436  000000
 9462
 9463  046440  001250  035634  035636  DT3:    .WORD   DRIVE,RD.ADR,RD.WRD,0
       046446  000000
 9464
 9465  046450  001250  036036  036034  DT4:    .WORD   DRIVE,WRT.AD,WRT.WD,RD.WRD,0
       046456  035636  000000
 9466
 9470  046462  037242  037252  037254  DT14.0: .WORD   BLK0+$RHCS1,BLK0+$RHCS2,BLK0+$RHDS1,BLK0+$RHER1
  (1)  046470  037256
  (1)  046472  037302  037304  037306          .WORD   BLK0+$RHER2,BLK0+$RHER3,BLK0+$RHEC1,BLK0+$RHEC2,0
  (1)  046500  037310  000000
  (1)  046504  037500  037510  037512  DT14.1: .WORD   BLK1+$RHCS1,BLK1+$RHCS2,BLK1+$RHDS1,BLK1+$RHER1
  (1)  046512  037514
  (1)  046514  037540  037542  037544          .WORD   BLK1+$RHER2,BLK1+$RHER3,BLK1+$RHEC1,BLK1+$RHEC2,0
  (1)  046522  037546  000000
  (1)  046526  037736  037746  037750  DT14.2: .WORD   BLK2+$RHCS1,BLK2+$RHCS2,BLK2+$RHDS1,BLK2+$RHER1
  (1)  046534  037752
  (1)  046536  037776  040000  040002          .WORD   BLK2+$RHER2,BLK2+$RHER3,BLK2+$RHEC1,BLK2+$RHEC2,0
  (1)  046544  040004  000000
  (1)  046550  040174  040204  040206  DT14.3: .WORD   BLK3+$RHCS1,BLK3+$RHCS2,BLK3+$RHDS1,BLK3+$RHER1
  (1)  046556  040210
  (1)  046560  040234  040236  040240          .WORD   BLK3+$RHER2,BLK3+$RHER3,BLK3+$RHEC1,BLK3+$RHEC2,0
  (1)  046566  040242  000000
  (1)  046572  040432  040442  040444  DT14.4: .WORD   BLK4+$RHCS1,BLK4+$RHCS2,BLK4+$RHDS1,BLK4+$RHER1
  (1)  046600  040446
  (1)  046602  040472  040474  040476          .WORD   BLK4+$RHER2,BLK4+$RHER3,BLK4+$RHEC1,BLK4+$RHEC2,0
  (1)  046610  040500  000000
  (1)  046614  040670  040700  040702  DT14.5: .WORD   BLK5+$RHCS1,BLK5+$RHCS2,BLK5+$RHDS1,BLK5+$RHER1
  (1)  046622  040704
  (1)  046624  040730  040732  040734          .WORD   BLK5+$RHER2,BLK5+$RHER3,BLK5+$RHEC1,BLK5+$RHEC2,0
  (1)  046632  040736  000000
  (1)  046636  041126  041136  041140  DT14.6: .WORD   BLK6+$RHCS1,BLK6+$RHCS2,BLK6+$RHDS1,BLK6+$RHER1
  (1)  046644  041142
  (1)  046646  041166  041170  041172          .WORD   BLK6+$RHER2,BLK6+$RHER3,BLK6+$RHEC1,BLK6+$RHEC2,0
  (1)  046654  041174  000000
  (1)  046660  041364  041374  041376  DT14.7: .WORD   BLK7+$RHCS1,BLK7+$RHCS2,BLK7+$RHDS1,BLK7+$RHER1
  (1)  046666  041400
  (1)  046670  041424  041426  041430          .WORD   BLK7+$RHER2,BLK7+$RHER3,BLK7+$RHEC1,BLK7+$RHEC2,0
  (1)  046676  041432  000000
 9471
 9475  046702  037244  037246  037250  DT15.0: .WORD   BLK0+$RHWC,BLK0+$RHBA,BLK0+$RHDA,BLK0+$RHAS,BLK0+$RHLA
  (1)  046710  037260  037262
  (1)  046714  037264  037266  037270          .WORD   BLK0+$RHDB,BLK0+$RHMR,BLK0+$RHDT,0
  (1)  046722  000000
  (1)  046724  037502  037504  037506  DT15.1: .WORD   BLK1+$RHWC,BLK1+$RHBA,BLK1+$RHDA,BLK1+$RHAS,BLK1+$RHLA
  (1)  046732  037516  037520
  (1)  046736  037522  037524  037526          .WORD   BLK1+$RHDB,BLK1+$RHMR,BLK1+$RHDT,0
  (1)  046744  000000
  (1)  046746  037740  037742  037744  DT15.2: .WORD   BLK2+$RHWC,BLK2+$RHBA,BLK2+$RHDA,BLK2+$RHAS,BLK2+$RHLA
  (1)  046754  037754  037756
```

```
   (1)  046760  037760  037762  037764          .WORD   BLK2+$RHDB,BLK2+$RHMR,BLK2+$RHDT,0
   (1)  046766  000000
   (1)  046770  040176  040200  040202  DT15.3: .WORD   BLK3+$RHWC,BLK3+$RHBA,BLK3+$RHDA,BLK3+$RHAS,BLK3+$RHLA
   (1)  046776  040212  040214
   (1)  047002  040216  040220  040222          .WORD   BLK3+$RHDB,BLK3+$RHMR,BLK3+$RHDT,0
   (1)  047010  000000
   (1)  047012  040434  040436  040440  DT15.4: .WORD   BLK4+$RHWC,BLK4+$RHBA,BLK4+$RHDA,BLK4+$RHAS,BLK4+$RHLA
   (1)  047020  040450  040452
   (1)  047024  040454  040456  040460          .WORD   BLK4+$RHDB,BLK4+$RHMR,BLK4+$RHDT,0
   (1)  047032  000000
   (1)  047034  040672  040674  040676  DT15.5: .WORD   BLK5+$RHWC,BLK5+$RHBA,BLK5+$RHDA,BLK5+$RHAS,BLK5+$RHLA
   (1)  047042  040706  040710
   (1)  047046  040712  040714  040716          .WORD   BLK5+$RHDB,BLK5+$RHMR,BLK5+$RHDT,0
   (1)  047054  000000
   (1)  047056  041130  041132  041134  DT15.6: .WORD   BLK6+$RHWC,BLK6+$RHBA,BLK6+$RHDA,BLK6+$RHAS,BLK6+$RHLA
   (1)  047064  041144  041146
   (1)  047070  041150  041152  041154          .WORD   BLK6+$RHDB,BLK6+$RHMR,BLK6+$RHDT,0
   (1)  047076  000000
   (1)  047100  041366  041370  041372  DT15.7: .WORD   BLK7+$RHWC,BLK7+$RHBA,BLK7+$RHDA,BLK7+$RHAS,BLK7+$RHLA
   (1)  047106  041402  041404
   (1)  047112  041406  041410  041412          .WORD   BLK7+$RHDB,BLK7+$RHMR,BLK7+$RHDT,0
   (1)  047120  000000
 9476
 9479  047122  037272  037274  037276  DT16.0: .WORD   BLK0+$RHSN,BLK0+$RHOF,BLK0+$RHCA,BLK0+$RHCC,BLK0+$STATUS,0
   (1)  047130  037300  037074  000000
   (1)  047136  037530  037532  037534  DT16.1: .WORD   BLK1+$RHSN,BLK1+$RHOF,BLK1+$RHCA,BLK1+$RHCC,BLK1+$STATUS,0
   (1)  047144  037536  037332  000000
   (1)  047152  037766  037770  037772  DT16.2: .WORD   BLK2+$RHSN,BLK2+$RHOF,BLK2+$RHCA,BLK2+$RHCC,BLK2+$STATUS,0
   (1)  047160  037774  037570  000000
   (1)  047166  040224  040226  040230  DT16.3: .WORD   BLK3+$RHSN,BLK3+$RHOF,BLK3+$RHCA,BLK3+$RHCC,BLK3+$STATUS,0
   (1)  047174  040232  040026  000000
   (1)  047202  040462  040464  040466  DT16.4: .WORD   BLK4+$RHSN,BLK4+$RHOF,BLK4+$RHCA,BLK4+$RHCC,BLK4+$STATUS,0
   (1)  047210  040470  040264  000000
   (1)  047216  040720  040722  040724  DT16.5: .WORD   BLK5+$RHSN,BLK5+$RHOF,BLK5+$RHCA,BLK5+$RHCC,BLK5+$STATUS,0
   (1)  047224  040726  040522  000000
   (1)  047232  041156  041160  041162  DT16.6: .WORD   BLK6+$RHSN,BLK6+$RHOF,BLK6+$RHCA,BLK6+$RHCC,BLK6+$STATUS,0
   (1)  047240  041164  040760  000000
   (1)  047246  041414  041416  041420  DT16.7: .WORD   BLK7+$RHSN,BLK7+$RHOF,BLK7+$RHCA,BLK7+$RHCC,BLK7+$STATUS,0
   (1)  047254  041422  041216  000000
 9480
 9481  047262     001     000     000  DF2:    .BYTE   1,0,0,0,0,0
       047265     000     000     000
 9482
 9483  047270     001     000     000  DF3:    .BYTE   1,0,0
 9484
 9485  047273     001     000     000  DF4:    .BYTE   1,0,0,0
       047276     000
 9486
 9487          047300                          .EVEN
 9488
 9489  047300  051120  051505  047105  LIN2C:  .ASCIZ  /PRESENT ORDER = /
       047306  020124  051117  042504
       047314  020122  020075     000
 9490  047321     040  050040  042522  LIN2P:  .ASCIZ  /  PREVIOUS ORDER = /
```

```
        047326  044526  052517  020123
        047334  051117  042504  020122
        047342  020075     000
 9491   047345     105  051122  051117  LIN2M:  .ASCIZ  /ERROR OCCURED DURING NON-DATA TRANSFER OPERATION/
        047352  047440  041503  051125
        047360  042105  042040  051125
        047366  047111  020107  047516
        047374  026516  040504  040524
        047402  052040  040522  051516
        047410  042506  020122  050117
        047416  051105  052101  047511
        047424  000116
 9492   047426  020052  051105  047522  LIN2S:  .ASCIZ  @* ERROR AT BAD TRACK/SECTOR@
        047434  020122  052101  041040
        047442  042101  052040  040522
        047450  045503  051457  041505
        047456  047524  000122
 9493   047462  051105  047522  020122  LINM3:  .ASCIZ  /ERROR AT C/
        047470  052101  041440     000
 9494   047475     040  000124          T:      .ASCIZ  / T/
 9495   047500  051120  051505  047105  LINN3:  .ASCIZ  /PRESENT ADDR = C/
        047506  020124  042101  051104
        047514  036440  041440     000
 9496   047521     040  000123          S:      .ASCIZ  / S/
 9497   047524  020040  050040  042522  LINP3:  .ASCIZ  /   PREV ADDR = C
        047532  020126  042101  051104
        047540  036440  041440     000
 9498   047545     123  040524  052122  LINS3:  .ASCIZ  /START CYL = /
        047552  041440  046131  036440
        047560  000040
 9499   047562  020040  047105  020104  LINEN3: .ASCIZ  /  END CYL = /
        047570  054503  020114  020075
        047576     000
 9500   047577     040  040440  052103  LINA3:  .ASCIZ  /  ACTUAL CYL = /
        047604  040525  020114  054503
        047612  020114  020075     000
 9501   047617     040  052040  045522  LINT3:  .ASCIZ  /  TRK = /
        047624  036440  000040
 9502   047630  051040  041510  020101  LINCA3: .ASCIZ  / RHCA = /
        047636  020075     000
 9503   047641     122  042110  020101  LINDA3: .ASCIZ  /RHDA = /
        047646  020075     000
 9504   047651     122  041110  020101  LINBA3: .ASCIZ  /RHBA = /
        047656  020075     000
 9505   047661     040  051040  053510  LINW3:  .ASCIZ  /  RHWC = /
        047666  020103  020075     000
 9506   047673     123  040524  052122  LINST3: .ASCIZ  /START TRK = /
        047700  052040  045522  036440
        047706  000040
 9507   047710  052123  051101  020124  LINSS3: .ASCIZ  /START SEC = /
        047716  042523  020103  020075
        047724     000
 9508   047725     102  043125  042506  LINM4:  .ASCIZ  /BUFFER ADDR = /
        047732  020122  042101  051104
```

```
                047740  036440  000040
 9509           047744  020040  044523  042532  LIN54:  .ASCIZ  /  SIZE = /
                047752  036440  000040
 9510           047756  020040  041501  052524  LINX4:  .ASCIZ  /  ACTUAL NMBR WRDS XFRD = /
                047764  046101  047040  041115
                047772  020122  051127  051504
                050000  054040  051106  020104
                050006  020075     000
 9511           050011     107  047517  020104  LIND5:  .ASCIZ  /GOOD DATA = /
                050016  040504  040524  036440
                050024  000040
 9512           050026  020040  040502  020104  LINB5:  .ASCIZ  /  BAD DATA = /
                050034  040504  040524  036440
                050042  000040
 9513           050044  020040  042523  052103  LINP5:  .ASCIZ  /  SECT POS = /
                050052  050040  051517  036440
                050060  000040
 9514           050062  042510  042101  051105  LINS5:  .ASCIZ  /HEADER FROM ERROR SECTOR = /
                050070  043040  047522  020115
                050076  051105  047522  020122
                050104  042523  052103  051117
                050112  036440  000040
 9515           050116  044122  041505  020061  LINEP5: .ASCIZ  /RHEC1 = /
                050124  020075     000
 9516           050127     040  044122  041505  LINE05: .ASCIZ  / RHEC2 = /
                050134  020062  020075     000
 9517           050141     123  041505  047524  LINB6:  .ASCIZ  /SECTOR IS ECC CORRECTABLE /
                050146  020122  051511  042440
                050154  041503  041440  051117
                050162  042522  052103  041101
                050170  042514  000040
 9518           050174  042523  052103  051117  LINC6:  .ASCIZ  /SECTOR READ CORRECTLY /
                050202  051040  040505  020104
                050210  047503  051122  041505
                050216  046124  020131     000
 9519           050223     103  051117  042522  LING6:  .ASCIZ  /CORRECTED ON /
                050230  052103  042105  047440
                050236  020116     000
 9520           050241     040  042522  051124  LINR6:  .ASCIZ  / RETRIES/
                050246  042511  000123
 9521           050252  047125  047503  051122  LINU06: .ASCIZ  /UNCORRECTABLE AFTER /
                050260  041505  040524  046102
                050266  020105  043101  042524
                050274  020122     000
 9522           050277     040  052040  052117  LIN7M:  .ASCIZ  /  TOTAL MISPOS ERR = /
                050304  046101  046440  051511
                050312  047520  020123  051105
                050320  020122  020075     000
 9523           050325     117  042122  051105  LIN70:  .ASCIZ  /ORDERS:/
                050332  035123     000
 9524           050335     040  047524  040524  LIN7P:  .ASCIZ  / TOTAL SEEKS = /
                050342  020114  042523  045505
                050350  020123  020075     000
 9525           050355     040  047524  040524  LIN7S:  .ASCIZ  / TOTAL SKI,OCYL ERR = /
```

```
                050362  020114  045523  026111
                050370  041517  046131  042440
                050376  051122  036440  000040
 9526           050404  020040  051105  047522  LIN7T:  .ASCIZ  /  ERRORS:/
                050412  051522  000072
 9527           050416  020040  051127  051504  LIN7X:  .ASCIZ  /  WRDS XFR:/
                050424  054040  051106  000072
 9528           050432  020040  051127  051504  LIN7R:  .ASCIZ  /  WRDS READ:/
                050440  051040  040505  035104
                050446     000
 9529           050447     104  043111  042506  LIN8M:  .ASCIZ  /DIFFERENT ERROR DURING RETRY/
                050454  042522  052116  042440
                050462  051122  051117  042040
                050470  051125  047111  020107
                050476  042522  051124  000131
 9530           050504  040504  040524  041440  LIN9B:  .ASCIZ  /DATA COMPARISON ERRORS/
                050512  046517  040520  044522
                050520  047523  020116  051105
                050526  047522  051522     000
 9531           050533     040  020040  020040  LIN9H:  .ASCII  /          GOOD    BAD/<15><12>
                050540  020040  020040  047507
                050546  042117  020040  020040
                050554  040502  006504     012
 9532           050561     114  041517  020040          .ASCIZ  /LOC      DATA    DATA/<15><12>
                050566  020040  020040  040504
                050574  040524  020040  020040
                050602  040504  040524  005015
                050610     000
 9533           050611     114  041517  020040  LIN9I:  .ASCIZ  /LOC      DATA/<15><12>
                050616  020040  020040  040504
                050624  040524  005015     000
 9534           050631     124  052117  046101  LIN9E:  .ASCIZ  /TOTAL COMPARE ERRORS = /
                050636  041440  046517  040520
                050644  042522  042440  051122
                050652  051117  020123  020075
                050660     000
 9535           050661     124  042510  042040  LIN9G:  .ASCIZ  /THE DATA COMPARED OK/<15><12>
                050666  052101  020101  047503
                050674  050115  051101  042105
                050702  047440  006513  000012
 9536           050710  051105  047522  020122  LIN10A: .ASCIZ  /ERROR BURST BEGINS AT WORD /
                050716  052502  051522  020124
                050724  042502  044507  051516
                050732  040440  020124  047527
                050740  042122  000040
 9537           050744  044440  020116  040504  LIN10B: .ASCIZ  / IN DATA FIELD OF ERROR SECTOR/<15><12>
                050752  040524  043040  042511
                050760  042114  047440  020106
                050766  051105  047522  020122
                050774  042523  052103  051117
                051002  005015     000
 9538           051005     105  051122  051117  LIN10C: .ASCII  /ERROR WAS NOT IN THE DATA READ - /<15><12>
                051012  053440  051501  047040
                051020  052117  044440  020116
```

```
                051026  044124  020105  040504
                051034  040524  051040  040505
                051042  020104  020055  005015
 9539           051050  041505  020103  047503          .ASCIZ  /ECC CORRECTION CAN'T BE PERFORMED/
                051056  051122  041505  044524
                051064  047117  041440  047101
                051072  052047  041040  020105
                051100  042520  043122  051117
                051106  042515  000104
 9540           051112  041505  020103  047503  LIN10H: .ASCII  /ECC CORRECTION RESULTS/<15><12>
                051120  051122  041505  044524
                051126  047117  051040  051505
                051134  046125  051524  005015
 9541           051142  042101  051104  020040          .ASCIZ  /ADDR    BAD     CORRECTED /<15><12>
                051150  020040  040502  020104
                051156  020040  020040  047503
                051164  051122  041505  042524
                051172  020104  005015     000
 9542           051177     103  047117  042524  LIN11H: .ASCIZ  /CONTENTS OF ERROR SECTOR (REPORTED ABOVE)/<15><12>
                051204  052116  020123  043117
                051212  042440  051122  051117
                051220  051440  041505  047524
                051226  020122  051050  050105
                051234  051117  042524  020104
                051242  041101  053117  024505
                051250  005015     000
 9543           051253     101  042104  020122          .ASCIZ  /ADDR    DATA/<15><12>
                051260  020040  042040  052101
                051266  006501  000012
 9544           051272  020040                  LIN4SP: .ASCII  /  /
 9545           051274     040                  LINSP:  .ASCII  / /
 9546           051275     040     000          LINSP0: .ASCIZ  / /
 9547           051277     125  044516  000124  UNTMSG: .ASCIZ  /UNIT/
 9548           051304  047440  043106  044514  UNTOFF: .ASCIZ  / OFFLINE/
                051312  042516     000
 9549           051315     040  047117  044514  UNTON:  .ASCIZ  / ONLINE/
                051322  042516     000
 9550           051325     040  047516  020124  UNTNOT: .ASCIZ  / NOT BEING TESTED/
                051332  042502  047111  020107
                051340  042524  052123  042105
                051346     000
 9551           051347     040  046101  042522  UNTASN: .ASCIZ  / ALREADY BEING TESTED/
                051354  042101  020131  042502
                051362  047111  020107  042524
                051370  052123  042105     000
 9552           051375     040  047516  020124  NOTRP:  .ASCIZ  / NOT AN RP04/
                051402  047101  051040  030120
                051410  000064
 9553           051412  047040  052117  050040  NOTPRS: .ASCIZ  / NOT PRESENT/
                051420  042522  042523  052116
                051426     000
 9554           051427     040  047125  040523  NOTSAF: .ASCIZ  / UNSAFE/
                051434  042506     000
 9555           051437     125  044516  020124  SYSTAT: .ASCIZ  /UNIT STATUS:/<15><12><12>
```

```
                051444  052123  052101  051525
                051452  006472  005012     000
 9556           051457     104  044522  042526  STATHD: .ASCII  /DRIVE PERFORMANCE SUMMARY/<15><12>
                051464  050040  051105  047506
                051472  046522  047101  042503
                051500  051440  046525  040515
                051506  054522  005015
 9557           051512  051104  020126  040520          .ASCIZ  /DRV PASS ORDERS   SEEKS   WRDS XFER   WRDS READ/
                051520  051523  047440  042122
                051526  051105  020123  020040
                051534  042523  045505  020123
                051542  020040  051127  051504
                051550  054040  042506  020122
                051556  020040  051127  051504
                051564  051040  040505  000104
 9558           051572  051440  043117  020124  STAHD1: .ASCIZ  / SOFT HARD  SKI MISP OTHER/<15><12>
                051600  040510  042122  020040
                051606  045523  020111  044515
                051614  050123  047440  044124
                051622  051105  005015     000
 9559           051627     104  047117  006505  PDONE:  .ASCIZ  /DONE/<15><12><12>
                051634  005012     000
 9560           051637     007  043077  052101  DROPNG: .ASCIZ  <07>/?FATAL OR EXCESSIVE ERRORS/
                051644  046101  047440  020122
                051652  054105  042503  051523
                051660  053111  020105  051105
                051666  047522  051522     000
 9561           051673     105  042116  047440  ENDPAS: .ASCIZ  /END OF PASS/
                051700  020106  040520  051523
                051706     000
 9562           051707     015  042412  042116  ENDTST: .ASCIZ  <15><12>/END OF TEST/
                051714  047440  020106  042524
                051722  052123     000
 9563           051725     015  042012  044522  DEASSG: .ASCIZ  <15><12>/DRIVE DEASSIGNED/
                051732  042526  042040  040505
                051740  051523  043511  042516
                051746  000104
 9564           051750  005015  025052  025052  DRNUM:  .ASCIZ  <15><12>/********** DRIVE #/
                051756  025052  025052  025052
                051764  042040  044522  042526
                051772  021440     000
 9565           051775     040  052123  051101  ASGND:  .ASCIZ  / STARTED/<15><12>
                052002  042524  006504  000012
 9566           052010  005015  050122  032060  TITLE:  .ASCII  <15><12>/RP04 MULTI-DRIVE EXERCISER PROGRAM/
                052016  046440  046125  044524
                052024  042055  044522  042526
                052032  042440  042530  041522
                052040  051511  051105  050040
                052046  047522  051107  046501
 9567           052054  005015  051450  047111          .ASCII  <15><12>@(SINGLE/DUAL PORT VERSION)@
                052062  046107  027505  052504
                052070  046101  050040  051117
                052076  020124  047526  051522
                052104  047511  020516
```

```
 9568  052110  005015  040515  047111          .ASCIZ  <15><12>/MAINDEC-11-DZRPN-B/<15><12><12>
       052116  042504  026503  030461
       052124  042055  051132  047120
       052132  041055  005015  000012
 9569  052140  047077  052117  042440  NOTNUF: .ASCIZ  /?NOT ENOUGH MEMORY/<15><12>
       052146  047516  043525  020110
       052154  042515  047515  054522
       052162  005015     000
 9570  052165     015  037412  023440  NEDCLK: .ASCIZ  <15><12>/? 'L' OR 'P' CLOCK REQUIRED ON SYSTEM/<15><12>
       052172  023514  047440  020122
       052200  050047  020047  046103
       052206  041517  020113  042522
       052214  052521  051111  042105
       052222  047440  020116  054523
       052230  052123  046505  005015
       052236     000
 9571  052237     056     000          PERIOD: .ASCIZ  /./
 9572  052241     077     000          QUES:   .ASCIZ  /?/
 9573  052243     007  050077  044522  PRTATN: .ASCIZ  <07>/?PRINTER REQUIRES MANUAL INTERVENTION/<15><12><12>
       052250  052116  051105  051040
       052256  050505  044525  042522
       052264  020123  040515  052516
       052272  046101  044440  052116
       052300  051105  042526  052116
       052306  047511  006516  005012
       052314     000
 9574  052315     111  053116  046101  INVLD:  .ASCIZ  /INVALID COMMAND/<15><12>
       052322  042111  041440  046517
       052330  040515  042116  005015
       052336     000
 9575  052337     0    042412  052116  ENTDAT: .ASCIZ  <15><12>/ENTER DATE: /
       052344  0511L    042040  052101
       052352  035105  000040
 9576  052356  047105  042524  020122  ENTID:  .ASCIZ  /ENTER OPERATOR I.D.: /
       052364  050117  051105  052101
       052372  051117  044440  042056
       052400  035056  000040
 9577  052404  005015  047105  042524  ENTDRV: .ASCIZ  <15><12>/ENTER I.D. FOR DRV #/
       052412  020122  027111  027104
       052420  043040  051117  042040
       052426  053122  021440     000
 9578  052433     072  000040          COLON:  .ASCIZ  /: /
 9579  052436  005015  040504  042524  DATEIS: .ASCIZ  <15><12>/DATE: /
       052444  020072     000
 9580  052447     015  047412  042520  IDIS:   .ASCIZ  <15><12>/OPERATOR I.D.: /
       052454  040522  047524  020122
       052462  027111  027104  020072
       052470     000
 9581  052471     015  005012  051104  HEDLIN: .ASCIZ  <15><12><12>/DRV  DRV I.D./<15><12>
       052476  020126  042040  053122
       052504  044440  042056  006456
       052512  000012
 9582  052514  047516  042516  005015  NONE:   .ASCIZ  /NONE/<15><12>
       052522     000
```

```
 9583  052523  000102  042101  052040  ENTADR: .ASCIZ  @BAD TRK/SEC ADRS FOR DRV #@
       052530  045522  051457  041505
       052536  040440  051104  020123
       052544  047506  020122  051104
       052552  020126  000043
 9584  052556  020077  047111  040526  BADENT: .ASCIZ  /? INVALID ENTRY/<15><12>
       052564  044514  020104  047105
       052572  051124  006531  000012
 9585  052600  005015  051120  043517  INTDON: .ASCIZ  <15><12>/PROGRAM INITIALIZATION COMPLETE/<15><12><12>
       052606  040522  020115  047111
       052614  052111  040511  044514
       052622  040532  044524  047117
       052630  041440  046517  046120
       052636  052105  006505  005012
       052644  000000
 9586          052646                  .EVEN
 9587
 9588  052646  053114                  PARLST: .WORD   PAR1             ;PARAMETER ENTRY LIST
 9589  052650  001272                          .WORD   MAXDL            ;LOCATION
 9590  052652  053124                          .WORD   PAR2             ;PARAMETER NAME
 9591  052654  001276                          .WORD   INTRVL
 9592  052656  053334                          .WORD   PAR19
 9593  052660  001270                          .WORD   PASCNT
 9594  052662  053134                          .WORD   PAR3
 9595  052664  001300                          .WORD   CMPLMT
 9596  052666  053144                          .WORD   PAR4
 9597  052670  001316                          .WORD   MAXCYL
 9598  052672  053154                          .WORD   PAR5
 9599  052674  001320                          .WORD   MINCYL
 9600  052676  053164                          .WORD   PAR6
 9601  052700  001322                          .WORD   MAXTRK
 9602  052702  053174                          .WORD   PAR7
 9603  052704  001324                          .WORD   MINTRK
 9604  052706  053204                          .WORD   PAR8
 9605  052710  001326                          .WORD   MAXSEC
 9606  052712  053214                          .WORD   PAR9
 9607  052714  001330                          .WORD   MINSEC
 9608  052716  053314                          .WORD   PAR17
 9609  052720  001334                          .WORD   BEGCOD
 9610  052722  053324                          .WORD   PAR18
 9611  052724  001332                          .WORD   BEGPAT
 9612  052726  177777                          .WORD   -1
 9613  052730  053304                          .WORD   PAR16
 9614  052732  001314                          .WORD   ENDET
 9615  052734  053224                          .WORD   PAR10
 9616  052736  001302                          .WORD   FORMAT
 9617  052740  053234                          .WORD   PAR11
 9618  052742  001162                          .WORD   $RPADR
 9619  052744  053244                          .WORD   PAR12
 9620  052746  001164                          .WORD   $RPVEC
 9621  052750  053254                          .WORD   PAR13
 9622  052752  001304                          .WORD   SAVLOD
 9623  052754  053264                          .WORD   PAR14
 9624  052756  001306                          .WORD   RATIO
```

```
                                                .WORD   PAR15
                                                .WORD   AUTOCK
                                                .WORD   PAR20
                                                .WORD   NOTPRT
                                                .WORD   0               ;TABLE TERMINATOR

052772  047105  042524  020122  ASKPAR: .ASCIZ  /ENTER PARAMETERS:  /
053000  040520  040522  042515
053006  042524  051522  020072
053014  000040

053016  005015  051105  047522  NGPAR:  .ASCIZ  <15><12>/ERROR IN PARAMETER:  /
053024  020122  047111  050040
053032  051101  046501  052105
053040  051105  020072  000040

053046  020055  042522  042455  REPAR:  .ASCIZ  /- RE-ENTER PARAMETERS/<15><12>
053054  052116  051105  050040
053062  051101  046501  052105
053070  051105  006523  000012

053076  020040  034050  020051  OCT8:   .ASCIZ  /  (8) /
053104      000

053105      040  030450  024460  DEC10:  .ASCIZ  / (10) /
053112  000040

053114  040515  042130  020114  PAR1:   .ASCIZ  /MAXOL  /
053122  000040

053124  047111  051124  046126  PAR2:   .ASCIZ  /INTRVL /
053132  000040

053134  046503  046120  052115  PAR3:   .ASCIZ  /CMPLMT /
053142  000040

053144  040515  041530  046131  PAR4:   .ASCIZ  /MAXCYL /
053152  000040

053154  044515  041516  046131  PAR5:   .ASCIZ  /MINCYL /
053162  000040

053164  040515  052130  045522  PAR6:   .ASCIZ  /MAXTRK /
053172  000040

053174  044515  052116  045522  PAR7:   .ASCIZ  /MINTRK /
053202  000040

053204  040515  051530  041505  PAR8:   .ASCIZ  /MAXSEC /
053212  000040

053214  044515  051516  041505  PAR9:   .ASCIZ  /MINSEC /
053222  000040

053224  047506  046522  052101  PAR10:  .ASCIZ  /FORMAT /
053232  000040

053234  051044  040520  051104  PAR11:  .ASCIZ  /$RPADR /
053242  000040

053244  051044  053120  041505  PAR12:  .ASCIZ  /$RPVEC /
053252  000040

053254  040523  046126  042117  PAR13:  .ASCIZ  /SAVLOD /
053262  000040

053264  040522  044524  020117  PAR14:  .ASCIZ  /RATIO  /
053272  000040

053274  052501  047524  045503  PAR15:  .ASCIZ  /AUTOCK /
053302  000040

053304  047105  042504  020124  PAR16:  .ASCIZ  /ENDET  /
```

```
                053312  000040
 9653           053314  042502  041507  042117  PAR17:  .ASCIZ  /BEGCOD /
                053322  000040
 9654           053324  042502  050107  052101  PAR18:  .ASCIZ  /BEGPAT /
                053332  000040
 9655           053334  040520  041523  052116  PAR19:  .ASCIZ  /PASCNT /
                053342  000040
 9656           053344  047516  050124  052122  PAR20:  .ASCIZ  /NOTPRT /
                053352  000040
 9657
 9658                                           .EVEN
 9659
 9660                                   ;*****************************************************************
 9661
 9662                                   .SBTTL  PATCH AREA
 9663
 9664                                   ;*****************************************************************
 9665
 9666           053354  000100                  .BLKW   100                     ;PATCH AREA
 9667
 9668                                   ;*****************************************************************
 9669
 9670           053354  000000  000000  000000  CYLDER: .WORD   0,0,0,0         ;HEADER BUFFER FOR 'READHD' ROUTINE
                053362  000000
 9671
 9672                   053364                  ENDPGM  =       .               ;LAST LOCATION OF PROG + 2
 9673
 9674                   000001                          .END
```

```
AGNRML  023616          4299    6826#
ACK   = 000123          3710#
ACTDRV  030542          7591#   7851*   7885*   8182*   8191*   8455
ACTSTR  030543          7597#   8457*   8472*
ASGN0   051775          4179    9565#
ASGN0   022052          6531#   6675    6681
ASGN1   022066          6535#
ASGN2   022122          6534    6543#   6546
ASGN3   022144          6540    6543    6548#   6571
ASGN4   022254          6547    6549    6568#
ASGN5   022264          6569    6572#
ASGN6   022272          6555    6574#
ASGN7   022302          6556    6576#
ASKPAR  052772          3951    9631#
ASNERR  023474          6541    6611    6629    6797#
ASNLST  041550          4186*   6302    6310    6538    6548    6598    6600*   6622    6632    6651    6811*   6882*   8996#
ASNMSG  023524          6536*   6574*   6583*   6585*   6587*   6610*   6628*   6803#
ATABIT  030612          7663#   7818    7911    8003    8318    8364    8392    8400    8408    8425    8529
ATTN    001252          3832#   7121*   9461
AUTOCK  001310          3832#   5439    5705    8230    9626
AVAIL   041624          4140    4180    4184*   4187    4211    4213    4245    4254    4255    4302    9002#
BADENT  052556          6753    9584#
BADPAR  002622          4011    4018    4022    4026    4030    4034    4038    4042    4046    4058    4062#
BEGCOD  001334          3832#   4047    4052    4053    4191    4192    9609
BEGPAT  001332          3832#   4043    4194    9611
BEGSIZ  001336          3832#   4012    4014*   4196    4197
BIT0  = 000001          3693#
BIT00 = 000001          3693#   5561    6596
BIT01 = 000002          3693#   5437    5584
BIT02 = 000004          3693#   5005    5556
BIT03 = 000010          3693#   4465    4816    8339    8644
BIT04 = 000020          3693#   4465    4474    8375
BIT05 = 000040          3693#   4917    7748    8132    8354    8503
BIT06 = 000100          3693#   4524    4525    4620    4635    7349    7825    7892    7940    8221    8699
BIT07 = 000200          3693#   4366    4477    7825    8077    8203    8339    8354    8375    8404
BIT08 = 000400          3693#   4480    7825    8516
BIT09 = 001000          3693#   4380    8501
BIT1  = 000002          3693#
BIT10 = 002000          3693#   4378    4492    4500    7468    8120
BIT11 = 004000          3693#   4376    4495    7794    7921    8094    8426
BIT12 = 010000          3693#   4374    4503    4524    4525    4554    4733    4737    4771    5017    5564    7790    7809
                        7825    7900    8365    8700
BIT13 = 020000          3693#   4483    4788    4884    7468    7882    8589
BIT14 = 040000          3693#   4353    4355    4382    4468    4471    4509    4658    4704    4746    4852    4867    7898
                        7918    8311    8442    8559    8596    8600    8697
BIT15 = 100000          3693#   4351    4509    4524    4599    4633    4884    7882    7898    7900    7918    8094    8120
                        8221    8339    8354    8365    8375    8442    8501    8516    8559    8566
BIT2  = 000004          3693#   4384    8566
BIT3  = 000010          3693#   4486
BIT4  = 000020          3693#   4710
BIT5  = 000040          3693#   4489    5896    8258
BIT6  = 000100          3693#   4554
BIT7  = 000200          3693#   4550    4752    5392
BIT8  = 000400          3693#   4663    4687    4694    4724    4731    4884
```

```
BIT9  = 001000  3693#  4498  4884
BLKADR  042034  3903  6558  6559  6602  6625  9012#
BLK0  037056  6301  6648  8980#  9015  9470  9475  9479
BLK1  037314  8980#  9015  9470  9475  9479
BLK2  037552  8980#  9015  9470  9475  9479
BLK3  040010  8980#  9015  9470  9475  9479
BLK4  040246  8980#  9015  9470  9475  9479
BLK5  040504  8980#  9015  9470  9475  9479
BLK6  040742  8980#  9015  9470  9475  9479
BLK7  041200  8980#  9015  9470  9475  9479
BPTVEC=  000014  3693#
BUFTBL  041712  3931*  3932*  3933*  3934*  3936*  3939*  3941*  3945  3947  3990*  3992*  4006  5458
  5461  5463  5479*  5495  5496  5508*  5513*  5514  5522  5523  5536*  9007#
CI1  032050  7934  7961#
CI2  032060  7963#
CI3  032156  7936  7984#
CI4  032264  7925  8005#
CI5  032642  7983  9004  8082#  8091
CI6  032664  8021  8031  8033  8035  8087#
CI7  032700  7888  7973  7977  7981  7989  7998  8002  8013  8020  8026  8030  8044  8050
  8054  8064  8075  9090  8092#  8177  8207  8388
CI7B  032716  8093  8096#  8389
CI8  032770  8099  8108#  8276  8551
CKADR  002260  4006#
CKBUS  010506  4358  4987#
CKCLK  020172  4076  6267#
CKCLK1  020266  6269  6280#
CKCLK2  020334  6281  6289#
CKCLK3  020352  6279  6288  6293#
CKERR  010406  4357  4966#
CKFMT  006536  4476  4687#
CKHCE  006736  4479  4724#
CKPAR  002326  4013  4015#
CKPTR  020124  4073  6255#
CKPTR1  020156  6256  6261#
CKPTR2  020162  6260  6262#
CLKFLG  001202  3832#  5878  6267*  6272*  6283*  6427
CLOCK  021344  6275  6285  6437#
CLRDPB  013204  5414#  6560
CLRQUE  036552  7730  8131  8526  8767#
CMCNT  011332  5014*  5015*  5024  5026  5029*  5086  5111#
CMCTL  011334  5016*  5017*  5032  5098*  5112#
CMDAT  011054  5031  5048  5052#
CMHED  010760  5032#
CMPAR  010572  4361  5005#
CMPARD  010622  4584  5012#
CMPLMT  001300  3832#  5019  9595
CMPRES  015234  4159  4226  4246  4256  4280  4306  4313  4344  5767#  5770
CMPRT  011340  5050  5064  5074  5084  5118#
CMPRX  011312  5045  5047  5087  5100#
CMSEC  011336  5018*  5035  5090*  5091  5093*  5113#
CMSTR  010702  5021#  5089  5092  5096  5099
CMTRK  011337  5094*  5095  5097*  5114#
COLON  052433  6692  9578#
```

```
COMTBL  042054          4193    5750    9018#
CONVEA  023344          6721    6729    6739    6764#
CYLDER  053554          4733*   4734    4737*   4771*   6001    6113    8984    9670#
DATAPK  022740          6518    6674#
DATAO   042622          9109    9112#
DATAI   042662          9094    9095    9096    9097    9098    9099    9100    9101    9102    9103    9104    9105    9106
                        9107    9108    9110    9117#
DATE    001226          3832#   3914*   3915*   3916*   3917*   6639    6642
DATEIS  052436          6641    9579#
DCKER   005410          4508    4516#
DCKER1  005426          4520#   4806
DEASGN  022364          6508    6593#
DEASSG  051725          4155    9563#
DEC10   053105          3977    9635#
DF2     047262          3845    3866    9481#
DF3     047270          3852    9483#
DF4     047273          3859    9485#
DH14    046145          5922    9451#
DH15    046251          5928    9454#
DH16    046350          5932    9457#
DH2     046003          3843    3864    9445#
DH3     046060          3850    9447#
DH4     046106          3857    9449#
DISPLA= 177570          3693#   7468*
DISPLY= 104422          4396    4426    4444    4455    4519    4588    4593    4601    4616    4654    4678    4700    4742
                        4768    4783    4805    4811    4829    4839    4848    4863    4879    4899    4910    4929    4941
                        4946    4947    4948    4956    4977    4995    5139    5144    5145    5146    5159    5162    5165
                        5172    5176    5229    5234    5266    5271    5274    5279    5281    5282    5316    5322    5327
                        5329    5409    5877    5880    5881    5882    5887    5888    5892    5898    5899    5905    5910
                        5911    5921    5922    5923    5924    5928    5932    5942    5944    5952    5954    5958    5965
                        5967    5971    5976    5981    5987    5992    5995    6000    6003    6007    6020    6023    6026
                        6031    6035    6036    6041    6042    6047    6054    6057    6058    6061    6066    6069    6072
                        6077    6088    6092    6101    6103    6108    6113    6114    6119    6122    6123    6126    6131
                        6132    6137    6140    6142    6147    6152    6156    6157    6162    6167    6172    6175    6180
                        6185    6186    6194    6199    6204    6207    6210    6211    6219    7072    7230    7480#
DONE    005114          4368    4465#
DPINT   030516          7551#   7757    7760    7836*   7861    7966    8253    8291    8432    8434*   8493    8531    9543
                        8553*
DPRQS   030526          7564#   7870    7944*   8294    8446*   8495    8533    8545    8562*
DRIVE   001250          3832#   7122*   9461    9463    9465
DRNUM   051750          4392    4414    4440    6817    6858    6862    9564#
DROP    023534          4402    4450    6809#   6829
DROPNG  031637          6816    9560#
DRVACT  030456          7511#   7878    8038*   8085*   8102*   8112    8123*   8196*   8263    8324    8337    8369*   8376*
                        8385    8399*   8504*   8510    8517*
DRVCLR= 000111          3705#
DRVER   006504          4512    4677#
DRVINT  031070          7750    7783#   7864    8416    8435    8508
DRVQUE  036654          7873    7890    8236    8807#
DRVSTA  030466          4086    6554    7525#   7740*   7741*   7742*   7743*   7767*   7784*   7827*   7829*   7835*   7857
                        7868    7896    7915    8039*   8251    8301    8309    8362*   8370*   8437    8554*
DRVTYP  030476          4094    4096    4098    6577    6579    6581    7538#   7787*   7800*
DTEER   007352          4505    4802#
DTUW    030610          6550    7657#   7736    7927    7982*   8103    8106*   8115    8129*   8185    8197*   8297    8498
```

```
                        8513    8524*   8535    8594
DT14.T  042476          5918    9060*
DT14.0  046462          9060    9470*
DT14.1  046504          9061    9470*
DT14.2  046526          9062    9470*
DT14.3  046550          9063    9470*
DT14.4  046572          9064    9470*
DT14.5  046614          9065    9470*
DT14.6  046636          9066    9470*
DT14.7  046660          9067    9470*
DT15.T  042516          5919    9069*
DT15.0  046702          9069    9475*
DT15.1  046724          9070    9475*
DT15.2  046746          9071    9475*
DT15.3  046770          9072    9475*
DT15.4  047012          9073    9475*
DT15.5  047034          9074    9475*
DT15.6  047056          9075    9475*
DT15.7  047100          9076    9475*
DT16.T  042536          5920    9078*
DT16.0  047122          9078    9479*
DT16.1  047136          9079    9479*
DT16.2  047152          908C    9479*
DT16.3  047166          9081    9479*
DT16.4  047202          9082    9479*
DT16.5  047216          9083    9479*
DT16.6  047232          9084    9479*
DT16.7  047246          9085    9479*
DT2     046422          3844    3865    9461*
DT3     046440          3851    9463*
DT4     046450          3858    9465*
DUNIT   041560          4133    6602*   6813*   6884*   8998*
ECBAD0  012512          5247*   5271    5291*
ECBAD1  012520          5261*   5279    5294*
ECBIT   012476          5221*   5222*   5223    5224*   5241    5243*   5285*
ECC     011736          4577    5205*
ECCX    012470          5273    5280    5282*
ECC1    012334          5254    5260    5266*
ECC2    012464          5238    5281*
ECGD    012510          5290*
ECGD1   012516          5293*
ECMSK0  012502          5239*   5244*   5249    5251    5287*
ECMSK1  012504          5240*   5245*   5253    5262    5264    5288*
ECSEC   012500          5205*   5214*   5217*   5219*   5235    5286*
ECWRD   012506          5223*   5225*   5226*   5227*   5228*   5230    5235*   5236    5247    5250    5251*   5252*   5255
                        5271    5289*
ECWRD1  012514          5248*   5255*   5256*   5257    5259*   5261    5263    5264*   5265*   5272    5279    5292*
EMPTYQ  036634          8100    8222    8377    8444    8567    8792*
EMTVEC= 000030          3693*   3893*   3896*
EM1     043622          3835    9378*
EM10    044112          4407    9388*
EM11    044155          4413    9390*
EM12    044210          4391    4396    9392*
EM13    044241          4426    9394*
```

```
EM14    044313          4439    4444    9396#
EM15    044335          4455    9398#
EM2     043672          3842    9380#
EM20    044371          4654    9400#
EM21    044412          4519    9402#
EM22    044443          4616    9404#
EM23    044516          4601    9406#
EM24    044575          4700    9408#
EM25    044643          4742    9410#
EM26    044724          4848    9412#
EM27    044751          4863    9414#
EM3     043717          3849    9382#
EM30    045006          4678    9416#
EM31    045040          4783    9418#
EM32    045103          4805    9420#
EM33    045136          4811    9422#
EM34    045213          4910    9424#
EM35    045255          4829    9426#
EM36    045313          4839    9428#
EM4     043775          3856    9384#
EM40    045344          4879    9430#
EM41    045402          4995    9432#
EM42    045446          5139    9434#
EM43    045523          4941    4946    9436#
EM44    045570          4977    9438#
EM5     044053          3863    9386#
EM50    045654          4899    9439#
EM51    045722          4768    9441#
EM60    045765          4929    9443#
ENDCMP  011614          5100    5170#
ENDCON  001260          3832#   6836    6839
ENDET   001314          3832#   6834    9614
ENDPAS  051673          6848    9551#
ENDPGM= 053564          3928    3932    3934    5548    9672#
ENDSEK  001264          3832#   6842    6845
ENDTST  051707          6857    9562#
ENTADP  052523          6704    9583#
ENTDAT  052337          3911    9575#
ENTDRV  052404          6686    9577#
ENTID   052356          3918    9576#
ENTPAR  002064          3955    3962#   4068
ENTPR   002162          3966    3986#   7168    7209
ERCTR   011326          4585    4958*   5012*   5049*   5060    5073*   5109#   5170    5173
ERENTR  025170          7121#   7468
ERPRC1  004472          4367    4372#   4594    5410
ERPROC  004456          4350    4354    4356    4366#
ERRVEC= 000004          3693#   6256*   6257*   6262*   6269*   6270*   6281*   6293*   7474*
FACTOR  013414          5431*   5432*   5434*   5435    5445    5453#
FALPAR  004664          4379    4412#
FALPR1  004674          4408    4414#
FILBUF  014004          4233    4267    5548#
FILLO   025043          7042*   7052    7086#
FMTER   007546          4696    4847#
FORMAT  001302          3832#   4055    4322    5718    9616
```

```
FRSTER  011321          4583*  4938   4961*  5011*  5076   5104*  5120   5135   5147*
GENDPB  041436          4560*  4564   5334*  5335*  5337   5345*  5346*  5348   5350   5356*  5357*  5359   5366*
                        5367*  5368*  5369*  5370*  5372   5374   6552*  8984*
GENREG  041456          4694   4731   8985   8986*
GETADR  023044          6562   6703*
GETBUF  013416          4230   4263   4544   5457*
GETDEC  025354          3978   7173*
GETID   022764          6561   6685*
GETOCT  025216          3983   7131*
GETPAR  015036          4228   5737*
GETPAT  015010          4335   5696   5726*  5731   5813
GETREG= 000141          3712*
GETREM  024110          5622   5638   5648   5658   5668   5704   5727   6889*
GETREQ  036730          7912   8117   8214   8305   8329   8387   8441   8500   8515   8556   8563   8828*
GETRQS  037040          8892*
GETRTS  037034          7765   7831   7884   7938   8134   8190   8471   8548   8684   8783   8891*
GNS     = ****** U      3722   7479   7480
GODRIV  014252          4234   4268   4546   4623   5385   5602*
HCEER   007624          4738   4862*
HCRCER  006354          4482   4651*  4689   4726
HEDLIN  052471          6647   9581*
HOLD    023470          6723   6725   6727   6732   6734   6736   6742   6744   6746   6764*  6773   6775*  6776
                        6777*  6778*  6779*  6784*  6786*  6792*
HOUR    021614          3907*  5880   6430   6459*  6460   6462   6464*  6465*  6467*  6479*
HZ      001214          3832*  3905   6440
IAEER   007466          4494   4828*
IDIS    052447          6645   9580*
IDLE    004062          4212   4257   4266   4285*
IDLE1   004170          4290   4308*
INCHRD  021174          4568   6420*
INCMIS  021244          4773   6422*
INCSKI  021220          4903   6421*
INCSOF  021150          4576   4582   4661   4708   4750   6419*
INCTOT  021270          4400   4417   4430   4448   4459   4569   4587   4592   4606   4641   4644   4662   4681
                        4709   4751   4774   4787   4815   4832   4841   4856   4871   4883   4902   4914   4932
                        4981   4999   5177   6423*
INTDON  052600          4116   9585*
INTRVL  001276          3832*  3910*  4001*  4067*  6450*  6470   6472   6474*  9591
INVLD   052315          6524   9574*
IOTVEC= 000020          3693*
ISR     033274          7745   8182*
KIPARO= ****** U        7474
KSR     021634          6492*  7373
LA      033136          7931   8147*
LACNT   030554          7616*  8155*  8156   8173*
LA30    001216          3832*  6331   6372
LIMIT   011330          5019*  5020*  5110*  5149   5151*
LINA3   047577          6000   9500*
LINB3   047651          6020   9504*
LINB5   050026          6092   9512*
LINB6   050141          6131   6147   9517*
LINCA3  047630          6058   9502*
LINC6   050174          6137   9518*
LINDA3  047641          6054   9503*
```

```
LIND5   050011          6088    9511#
LINEN3  047562          5995    9499#
LINE05  050127          6123    9516#
LINEP5  050116          6119    9515#
LINE1   015714          4395    4425    4443    4454    4518    4598    4653    4677    4699    4741    4767    4782    4804
                        4810    4828    4838    4847    4862    4878    4898    4909    4928    4940    4976    4994    5138
                        5874#
LINE2   015760          4397    4427    4445    4458    4520    4605    4619    4655    4679    4701    4743    4769    4784
                        4812    4830    4840    4849    4864    4880    4900    4911    4930    4942    4978    4996    5140
                        5837#   6220
LINE2A  016104          5997    5900    5908#
LINE2B  016122          5909    5912#
LINE3   016332          4398    4428    4446    4458    4521    4605    4619    4656    4680    4702    4744    4785    4850
                        4865    4881    4931    4979    5952#
LINE3A  016340          4912    4943    5141    5954#
LINE3B  016512          4901    5986#
LINE3C  016524          4770    5989#
LINE3D  016646          4997    6018#
LINE3E  016714          4813    6031#
LINE3F  017016          4831    6052#
LINE4   017070          4399    4429    4447    4458    4522    4605    4619    4657    4703    4745    4786    4814    4851
                        4866    4882    4913    4944    4980    4998    5142    6066#
LINE5   017170          4605    4619    4660    4706    4748    4854    4869    6088#
LINE5A  017270          4707    4749    4772    4855    4870    6108#
LINE5B  017362          4575    6119#
LINE6   017424          4527    6131#
LINE6A  017436          4581    6137#
LINE6B  017464          4574    6147#
LINE6C  017474          4610    4628    4637    4667    4714    4756    4792    4820    4890    4920    6152#
LINE6D  017524          4579    4612    4670    4717    4759    4795    4823    4892    4923    6162#
LINE7   017534          4401    4431    4445    4460    4570    4589    4642    4645    4668    4671    4682    4715    4718
                        4757    4760    4793    4796    4821    4833    4842    4857    4872    4893    4921    4933    4982
                        5000    5178    5406    6167#
LINE7A  017662          4775    4904    6194#
LINE8   020002          4567    4639    5405    6219#
LING6   050223          6152    9519#
LINKDV  024136          5210    5305    5690    6099    6892    6902#
LINKML  024406          6974#
LINM3   047462          5952    9493#
LINM4   047725          6066    9508#
LINN3   047500          5954    9495#
LINP3   047524          5967    9497#
LINP5   050044          6101    9513#
LINR6   050241          6156    9520#
LINSP   051274          4956    5159    5162    5271    5279    5322    5924    5942    6035    6041    6057    6113    6122
                        6346    6350    6357    6364    6376    9545#
LINSP0  051275          6371    6383    6706    9546#
LINSS3  047710          6042    9507#
LINST3  047673          6036    9506#
LINS3   047545          5992    6031    9198#
LINS4   047744          6069    9509#
LINS5   050062          6108    9514#
LINT3   047617          6003    9501#
LINU06  050252          6162    9521#
```

```
LINW3   047661          6023    9505#
LINX4   047756          6072    9510#
LIN10A  050710          5229    9536#
LIN10B  050744          5234    9537#
LIN10C  051005          5281    9538#
LIN10H  051112          5266    9540#
LIN11H  051177          5316    9542#
LIN2C   047300          5888    9489#
LIN2M   047345          5899    9491#
LIN2P   047321          5892    9490#
LIN2S   047426          5911    9492#
LIN3.1  016344          5953    5955#
LIN3.2  016626          5994    5997    6002    6006    6010#   6071    6076    6102    6155    6174    6206    6209
LIN3.3  016534          5986    5989    5992#
LIN3.4  016566          5990    6000#
LIN4SP  051272          4085    6336    6375    6656    9544#
LIN4.1  017150          6022    6025    6068    6080#
LIN6.1  017442          6138#   6148
LIN6.2  017500          6153#   6163
LIN7A   050277          6204    9522#
LIN7O   050325          6167    6194    9523#
LIN7P   050335          6199    9524#
LIN7R   050432          6180    9528#
LIN7S   050355          6207    9525#
LIN7T   050404          6172    9526#
LIN7X   050416          6175    9527#
LIN8M   050447          4593    5409    6219    9529#
LIN9B   050504          5144    9530#
LIN9E   050631          5172    9534#
LIN9G   050661          4588    9535#
LIN9H   050533          5146    9531#
LIN9I   050611          4948    9533#
LKPAR   002036          3943    3946    3951#   3958
MAIN    003120          4117    4125#   4314    5800
MAIN1   003264          4139    4164#
MAIN2   003534          4174    4209#   4247    4281
MAIN3   003762          4210    4261#
MATCH   011660          5052    5185#
MAXCYL  001316          3832#   4015    4019    5654    5788    9597
MAXDL   001272          3832#   3942    3944*   3945    3947*   4006    4008    4012    4014    4331    4333    5552    5665
                        5693    5695    5801    5802    9589
MAXER   001274          3832#   4002*   6828
MAXSEC  001326          3832#   4031    4035    5634    9605
MAXTRK  001322          3832#   4023    4027    5644    5791    9601
MCPEMX  030622          7675#   8580
MEMERR  020014          3930    6231#
MINCYL  001320          3832#   4019    4190    5655    5659    5795    9599
MINSEC  001330          3832#   4035    4188    5635    5639    5794    9607
MINTRK  001324          3832#   4027    4189    5645    5649    5793    9603
MINUTE  021620          3908*   5881    6431    6451*   6452    6454*   6455*   6456    6458*   6481#
MNDLTA  030636          7687#   8171
MXDLTA  030634          7685#   8169
MXLACT  030632          7683#   8156
MXWNDW  030640          7689#   7991
```

L15

DZRPNB.P11 CROSS REFERENCE TABLE

```
M.DPID  024230            6909    6923#
M.DPIM  024470            6979    6993#
M.DP01  024500            6996#   7005
M.DP02  024520            7000    7004#
M.DP40  024266            6934#   6955
M.DP41  024322            6939    6946#
M.DP42  024332            6945    6950#
M.DP44  024364            6957    6962#
M.DP50  024376            6932    6966#
NEDCLK  052165            6290    9570#
NEWASN  022046            6504    6530#
NEWUNT  041602            4168    4184    4185*   6558*   9000#
NGPAR   053016            4063    9632#
NOMTCH  010174            4938#   5054
NONE    052514            6659    9582#
NOTNUF  052140            6232    9569#
NOTPRS  051412            4102    6585    9553#
NOTPRT  001312            3832#   5859    9628
NOTRP   051375            4100    6583    9552#
NOTSAF  051427            4104    6587    9554#
NRML    023640            4300    6834#
NRMLX   024106            6841    6844    6846    6878    6885#
NRML1   023672            6835    6838    6842#
NRML2   023714            5796    6837    6840    6843    6847#
OCNT    025042            7044*   7073*   7085#
OCT8    053076            3982    9634#
OFFCOD  042142            4560    9032#
OFFSET= 000115            3707#   5357
OFFST   012752            4563    5356#
OFLIN   004772            4383    4438#
OFMSG0  042170            9040    9048#
OFMSG1  042223            9041    9049#
OFMSG2  042257            9042    9050#
OFMSG3  042313            9043    9051#
OFMSG4  042347            9044    9052#
OFMSG5  042403            9045    9053#
OFMSG6  042440            9046    9054#
OFMTBL  042152            6139    9040#
OMODE   025044            7043*   7048    7051*   7062*   7087#
OPERID  001240            3832#   3921*   3922*   3923*   6643    6646
OPIER   007242            4485    4780#
OPIER1  007306            4789#
OPMNEM  042062            5904    9025#
OPT     031640            7880    7909#   8217    8409    8447
ORDERQ  041526            3901    4235    4269    4288    8994#
PACK    001224            3832#   3998*   6530*   6564    6566    6674*   6680*
PARER   007374            4488    4810#
PARLST  052646            3964    9588#
PARQ    041670            4220    4241    4275    4308    4310    4312    4338    9005#
PAR1    053114            4010    9588    9637#
PAR10   053224            9615    9646#
PAR11   053234            9617    9647#
PAR12   053244            9619    9648#
PAR13   053254            9621    9649#
```

PAR14 053264 4041 9623 9650#
PAR15 053274 9625 9651#
PAR16 053304 9613 9652#
PAR17 053314 4057 9608 9653#
PAR18 053324 4045 9610 9654#
PAR19 053334 9592 9655#
PAR2 053124 9590 9638#
PAR20 053344 9627 9656#
PAR3 053134 9594 9639#
PAR4 053144 4017 9596 9640#
PAR5 053154 4021 9598 9641#
PAR6 053164 4025 9600 9642#
PAR7 053174 4029 9602 9643#
PAR8 053204 4033 9604 9644#
PAR9 053214 4037 9606 9645#
PASCNT 001270 3832# 6855 9593
PC =%000007 3693# 3927# 3978# 3983# 4073# 4076# 4077# 4115# 4131# 4156# 4159# 4218# 4226#
4227# 4228# 4230# 4233# 4234# 4246# 4256# 4263# 4267# 4268# 4280# 4295# 4297#
4299# 4300# 4306# 4307# 4311# 4313# 4327# 4335# 4344# 4345# 4357# 4358# 4361#
4362# 4386# 4395# 4397# 4398# 4399# 4400# 4401# 4417# 4421# 4425# 4427# 4428#
4429# 4430# 4431# 4434# 4443# 4445# 4446# 4447# 4448# 4449# 4454# 4458# 4459#
4460# 4461# 4502# 4516# 4517# 4518# 4520# 4521# 4522# 4523# 4527# 4531# 4534#
4544# 4546# 4563# 4567# 4568# 4569# 4570# 4571# 4572# 4573# 4574# 4575# 4576#
4577# 4579# 4581# 4582# 4584# 4587# 4589# 4590# 4591# 4592# 4598# 4605# 4606#
4608# 4610# 4612# 4614# 4615# 4619# 4623# 4628# 4637# 4639# 4641# 4642# 4643#
4644# 4645# 4646# 4647# 4651# 4652# 4653# 4655# 4656# 4657# 4660# 4661# 4662#
4665# 4667# 4668# 4669# 4670# 4671# 4672# 4673# 4677# 4679# 4680# 4681# 4682#
4683# 4692# 4693# 4697# 4698# 4699# 4701# 4702# 4703# 4706# 4707# 4708# 4709#
4712# 4714# 4715# 4716# 4717# 4718# 4719# 4720# 4729# 4730# 4739# 4740# 4741#
4743# 4744# 4745# 4748# 4749# 4750# 4751# 4754# 4756# 4757# 4758# 4759# 4760#
4761# 4762# 4766# 4767# 4769# 4770# 4772# 4773# 4774# 4775# 4776# 4780# 4781#
4782# 4784# 4785# 4786# 4787# 4790# 4792# 4793# 4794# 4795# 4796# 4797# 4798#
4802# 4803# 4804# 4810# 4812# 4813# 4814# 4815# 4818# 4820# 4821# 4822# 4823#
4828# 4830# 4831# 4832# 4833# 4834# 4838# 4840# 4841# 4842# 4843# 4847# 4849#
4850# 4851# 4854# 4855# 4856# 4857# 4858# 4862# 4864# 4865# 4866# 4869# 4870#
4871# 4872# 4873# 4874# 4878# 4880# 4881# 4882# 4883# 4888# 4890# 4892# 4893#
4894# 4898# 4900# 4901# 4902# 4903# 4904# 4905# 4909# 4911# 4912# 4913# 4914#
4915# 4918# 4920# 4921# 4922# 4923# 4928# 4930# 4931# 4932# 4933# 4934# 4940#
4942# 4943# 4944# 4956# 4962# 4976# 4978# 4979# 4980# 4981# 4982# 4983# 4994#
4996# 4997# 4998# 4999# 5000# 5001# 5007# 5010# 5034# 5037# 5040# 5043# 5050#
5051# 5052# 5054# 5064# 5074# 5084# 5100# 5101# 5122# 5123# 5128# 5129# 5133#
5134# 5138# 5140# 5141# 5142# 5148# 5155# 5158# 5161# 5164# 5166# 5174# 5175#
5177# 5178# 5179# 5201# 5210# 5232# 5233# 5271# 5279# 5283# 5305# 5319# 5324#
5330# 5339# 5352# 5361# 5379# 5385# 5404# 5405# 5406# 5408# 5427# 5451# 5471#
5518# 5558# 5583# 5589# 5614# 5618# 5622# 5625# 5638# 5648# 5658# 5668# 5671#
5680# 5684# 5696# 5699# 5704# 5722# 5727# 5730# 5733# 5744# 5763# 5771# 5775#
5784# 5796# 5813# 5816# 5822# 5825# 5842# 5864# 5883# 5891# 5895# 5907# 5925#
5931# 5934# 5936# 5941# 5945# 5956# 5957# 5961# 5962# 5963# 5964# 5966# 5969#
5970# 5974# 5975# 5979# 5980# 5982# 5986# 5988# 5989# 5994# 5997# 5998# 6002#
6006# 6008# 6011# 6012# 6014# 6022# 6025# 6027# 6033# 6034# 6039# 6040# 6045#
6046# 6048# 6056# 6060# 6062# 6068# 6071# 6076# 6078# 6081# 6082# 6084# 6091#
6094# 6099# 6102# 6104# 6113# 6115# 6121# 6125# 6127# 6133# 6143# 6155# 6158#
6170# 6171# 6174# 6178# 6179# 6183# 6184# 6190# 6197# 6198# 6202# 6203# 6206#
6209# 6215# 6220# 6221# 6251# 6263# 6294# 6309# 6316# 6322# 6326# 6339# 6348#

```
                        6349* 6357* 6364* 6371* 6383* 6388* 6389* 6391* 6396* 6401* 6403* 6419* 6420*
                        6421* 6422* 6423* 6433* 6504* 6508* 6512* 6518* 6522* 6526* 6540* 6542* 6543*
                        6545* 6553* 6560* 6561* 6562* 6573* 6575* 6589* 6611* 6612* 6626* 6629* 6638*
                        6670* 6699* 6756* 6805* 6822* 6830* 6865* 6885* 6892* 6896* 6909* 6914* 6965*
                        6968* 6979* 6983* 7007* 7035* 7082* 7094* 7109* 7114* 7125* 7126* 7167* 7208*
                        7234* 7270* 7277* 7282* 7325* 7330* 7350* 7359* 7373* 7468* 7469* 7473* 7474*
                        7475* 7476* 7477* 7478* 7730* 7759* 7766* 7792* 7880* 7888* 7894* 7912* 7917*
                        7925* 7934* 7936* 7939* 7942* 7961* 8045* 8076* 8080* 8081* 8086* 8095* 8100*
                        8107* 8117* 8121* 8131* 8133* 8135* 8177* 8187* 8189* 8212* 8213* 8214* 8217*
                        8222* 8223* 8245* 8267* 8268* 8279* 8290* 8305* 8306* 8307* 8316* 8329* 8345*
                        8363* 8377* 8387* 8391* 8403* 8407* 8409* 8418* 8441* 8443* 8444* 8447* 8500*
                        8502* 8515* 8526* 8551* 8556* 8563* 8567* 8568* 8584* 8604* 8606* 8638* 8685*
                        8706* 8784* 8796* 8834* 8851*
PCLOCK  001200          3832# 6268* 6273*
PDONE   051627          9559#
PERIOD  052237          9571#
PIRQ  = 177772          3693#
PIRQVE= 000240          3693#
POPQUE  036752          7917  7961  8076  8403  8843#
POSER   007166          4736  4766#
PRESET= 000121          3709#
PREVST  015536          5744  5784  5820#
PROCES  004372          4297  4349#
PRTATN  052243          7312  9573#
PRTBAD  012522          4523  4571  4915  5298#
PRTIM   005056          4385  4454#
PRTOCT  024652          4956  5158  5161  5164  5271  5279  5319  5324  5941  6056  6060  6091  6094
                        6113  6121  6125  7042#
PRTPB   016330          4298* 5861* 5908  5948# 6419  6420  6421  6422  6423
PR0   = 000000          3693#
PR1   = 000040          3693#
PR2   = 000100          3693#
PR3   = 000140          3693#
PR4   = 000200          3693#
PR5   = 000240          3693#
PR7   = 000340          3693#
PS    = 177776          3693# 3893* 3997* 7091  7092* 7104* 7108  7113  7474  7728  7729* 7756* 7764*
                        7849  7850* 7886* 7910  7937* 8162* 8483  8484* 8549*
PSW   = 177776          3693#
PUNSAF  004566          4375  4390#
PWRVEC= 000024          3693#
QCNT    036260          8709# 8768  8792* 8807  8809* 8829  8843*
QDRV0   036352          8720  8731  8740  8752#
QDRV1   036372          8721  8732  8741  8753#
QDRV2   036412          8722  8733  8742  8754#
QDRV3   036432          8723  8734  8743  8755#
QDRV4   036452          8724  8735  8744  8756#
QDRV5   036472          8725  8736  8745  8757#
QDRV6   036512          8726  8737  8746  8758#
QDRV7   036532          8727  8738  8747  8759#
QINPT   036270          8720# 8794  8811* 8812* 8813  8815*
QOUTPT  036310          8731# 8794* 8832  8845  8846* 8847  8849*
QSTART  036330          8740# 8774  8779  8815  8849
QSTOP   036332          8741# 8813  8847
```

```
QTERM = 036552          8748    8760#
QUES    052241          6799    9572#
RANWRT  014704          5625    5703#
RATIO   001306          3832#   4039    5623    9624
RDCHR = 104412          7414    7479#
RDDAT = 000171          3719#   9022
RDHD  = 000173          3720#   5370    5808    9023
RDLIN = 104414          3912    3919    3952    6693    6716    7132    7174    7479#
RD.ADR  035634          8582*   8583*   8585#   9463
RD.RP   035610          7796    7813    7819    8024    8048    8062    8149    8163    8205    8274    8331    8341    8356
                        8488    8537    8580#   8628    8641    8647
RD.RP1  035632          8584#   8608
RD.RP2  035734          8580*   8607#
RD.RP3  035740          8599    8609#
RD.WRD  035636          8586#   8587    9463    9465
READDR  020036          4327    4534    4692    4729    5822    5842    5961    6239#
READHD  012776          4693    4730    5366#
RECAL = 000107          3704#   5346
RECALT  012720          4766    5345#
RECALO  012724          5346#   6553
REDAPK  022752          6522    6680#
REFMT   004222          4318#   4572    4590    4646    4672    4719    4761    4797    4873
REFMTX  004370          4319    4321    4323    4340    4345#
REL   = 000113          3706#
RELBUF  013556          4307    4531    5494#
REPAR   053046          4066    9633#
RESREG= 104420          7475    7477    7479#
RESVEC= 000010          3693#
RETRY   013060          4608    4665    4712    4754    4790    4818    4888    4918    5385#   5403
RHAS  = 000016          7700#   7818*   8266*   8270    8318*   8354*   8392*   8408*   8425*   8528
RHBA  = 000004          7695#
RHCA  = 000034          7707#   7976    7988    8019
RHCC  = 000036          7708#   8150
RHCS1 = 000000          7693#   7807    7980    8001    8043    8063    8074    8089    8098    8206    9275    8348    9538
                        8629
RHCS2 = 000010          7697#   7748*   7788*   7790    7872*   7965*   7985*   8006*   8132*   8148*   8204*   8258*   8330*
                        8411*   8424*   8497*   8503*   8667*   8674    8695*   8700
RHDA  = 000006          7696#   7972    7997    8012
RHDB  = 000022          7702#   8671
RHDS1 = 000012          7698#   7814    7834*   7920*   8332    8357    8412    8489
RHDT  = 000026          7704#   7797
RHEC1 = 000044          7711#
RHEC2 = 000046          7712#   8682
RHER1 = 000014          4694    4731    7699#   7820    8342    8413    8642
RHER2 = 000040          7709#   8414
RHER3 = 000042          7710#   8415
RHLA  = 000020          7701#   8164
RHMR  = 000024          7703#
RHOF  = 000032          7706#   7811    8025    8029    8049    8053
RHSN  = 000030          7705#
RHWC  = 000002          7694#
RNOP  = 000101          3701#
RPADR   030624          4074*   7679#   7747    7856    7863    7964    7984    8005    8109    8147    8194    8486    8582
                        8588    8602    8637    8666    8694
```

DZRPNB.P11 CROSS REFERENCE TABLE

```
RPERRS  C30446          7501#  7731   8311   8314   8412*  8413*  8414*  8415*  9461
RPINIT  030642          4077   7727#
RPTMR   035036          6469   8455#
RPVEC   030626          4075*  7680#  7744   7746   7850   8484
RPZERO  024564          7012   7014   7016   7018#
RPZRC   024650          7011*  7013*  7015*  7017*  7030   7037#
RP04    031352          5336   5347   5358   5371   5604   7849#
RP6     = 000300        3693#
RSTART  002214          3898   3957   3987   3991   3996#
RTC     = 000117        3708#  5335
RTNCTR  012674          5334#
R0      =%000000        3693#  4154*  4187*  4188*  4189*  4190*  4191*  4193*  4194*  4195*  4196*  4197*  4198*
                        4199*  4200   4202*  4213*  4214   4216   4221   4231*  4238   4239   4244   4261*  4264*
                        4272   4273   4278   4289*  4291   4310*  4324*  4328*  4329*  4330*  4333*  4334*  4336*
                        4337*  4341   4349   4351   4353   4355   4366   4373   4374   4376   4378   4380   4382
                        4384   4465   4468   4471   4474   4477   4480   4483   4486   4489   4492   4495   4498
                        4500   4503   4506   4509   4524*  4525   4529*  4535*  4536*  4537*  4538   4540*  4541*
                        4542*  4545*  4547   4550   4552   4554   4556*  4557*  4562*  4599   4602*  4607*  4620
                        4622*  4624   4627*  4630*  4631*  4633   4635   4658   4663*  4664*  4687   4704   4710*
                        4711*  4724   4734   4746   4752*  4753*  4788*  4789*  4816*  4817*  4852   4867   4884
                        4886*  4887*  4916*  4917*  4966   4968   4970   4972   4974   4987   4989   4991   4992
                        5005   5013   5014   5015   5016   5018   5024   5028   5029   5030   5088   5205   5206
                        5207   5209   5215   5221   5236   5239   5257   5300   5301   5302   5304   5312   5320
                        5334   5336*  5345   5347*  5356   5358*  5368   5369   5371*  5386   5389*  5392   5394
                        5396   5399   5401*  5402*  5415   5431   5432   5435*  5436*  5437   5441   5443*  5444*
                        5445*  5446*  5447   5449*  5450*  5464   5473   5475*  5476   5477*  5498   5500   5506
                        5507   5511   5512   5519   5520   5548   5552   5556   5559   5560   5561   5563   5565
                        5571   5584   5586   5587   5590*  5591*  5602   5603   5604*  5607*  5608*  5609*  5610
                        5612*  5613*  5630*  5640*  5650*  5660*  5661   5673*  5677   5686   5692*  5695*  5697*
                        5698*  5712   5714*  5715*  5721*  5738*  5739   5741*  5742*  5745*  5748*  5749   5750*
                        5751*  5752*  5753*  5754*  5755*  5756*  5757*  5758   5760*  5761*  5776   5778   5780
                        5783*  5785*  5786*  5787*  5788   5791   5793*  5794*  5795*  5801*  5802*  5803*  5804
                        5806*  5807*  5808*  5810*  5811*  5812*  5814*  5815*  5823*  5824*  5836   5838   5890
                        5894   5896   5914   5955   5968   5973   5978   5993   5996   6005   6021   6024   6032
                        6038   6044   6055   6059   6067   6070   6073   6074   6089*  6090   6093   6095   6096
                        6098   6120   6124   6154   6169   6173   6177   6182   6196   6201   6205   6208   6240
                        6245   6250   6298   6301*  6305*  6312*  6314   6317*  6321*  6347   6384   6387   6395
                        6398   6399   6402*  6419*  6420*  6421*  6422*  6423*  6437   6469*  6476*  6492   6525*
                        6559*  6563*  6566*  6625*  6695*  6696*  6697*  6710   6810   6813   6819   6828   6836
                        6839   6842   6845   6849   6852   6855   6868   6877*  6881   6884   6902   6907*  6912*
                        6930*  6931*  6937*  6942*  6944*  6947*  6949*  6959*  6960*  6974   6982*  6993*  6996*
                        7002*  7003*  7018   7019*  7020   7022*  7023*  7025   7027*  7028*  7033*  7133*  7134
                        7135   7137   7141   7160*  7175*  7176   7177   7179   7182   7201*  7222   7223*  7224
                        7226   7228*  7229   7232*  7262   7263*  7264   7267*  7316   7317*  7319   7322*  7469*
                        7471*  7472*  7473*  7474*  7475*  7477*  7479*  7727*  7750*  7765*  7767*  7783*  7796*
                        7806*  7810*  7813*  7819*  7830   7831*  7832*  7852*  7853   7864*  7873*  7883   7884*
                        7887*  7890*  7909*  7931*  7938*  7971*  7975*  7979*  7987*  7996*  8000*  8011*  8018*
                        8024*  8028*  8042*  8048*  8052*  8062*  8073*  8088*  8097*  8108*  8134*  9149*  9163*
                        8174   8175   8176*  8178*  8183*  8190*  8205*  8236*  8274*  8331*  8341*  8347*  8356*
                        8416*  8435*  8458*  8465*  8471*  8474*  8485*  8488*  8508*  8537*  8542   8548*  8550*
                        8583   8591   8592*  8609*  8611*  8624   8628*  8635*  8636*  8641*  8646   8647*  8652*
                        8653*  8654   8655*  8663*  8684*  8767*  8783*  8817   8818*  8879*  8898*
R1      =%000001        3693#  4140*  4141   4143   4145*  4147*  4148   4150   4152*  4154   4158*  4192*  4193
                        4220*  4221   4223   4245*  4255*  4279*  4288*  4289   4293*  4294   4301*  4305   4312*
```

```
                4338*  4339   4341   4343*  4530*  4559*  4560   4956   4957*  5013*  5032   5035   5038
                5041   5058   5070   5079   5125   5126*  5132*  5156   5163   5185   5187*  5193   5200*
                5300*  5309*  5311*  5314*  5318   5320   5323   5457   5463*  5464   5468*  5470*  5472
                5473*  5476*  5483   5485*  5486*  5494   5495*  5501   5503*  5506*  5507*  5511*  5512*
                5517*  5519*  5520*  5527   5535   5537   5539*  5540*  5559*  5563*  5564*  5565*  5566*
                5567*  5575*  5592*  5595*  5767*  5769*  5834   5835*  5836*  5838   5843   5845   5849
                5853*  5854   5863*  6138*  6139   6274*  6275*  6276*  6284*  6285*  6286*  6492   6525*
                6703   6709*  6710*  6711*  6714*  6727*  6736*  6746*  6749*  6751*  6752*  6755*  6902
                6908*  6910   6912*  6929*  6936*  6941*  6946*  6958*  6974   6982*  6994*  6997*  7001*
                7122   7139*  7147*  7149*  7151*  7154*  7157   7182*  7184   7186   7194*  7195   7469*
                7471*  7472*  7473*  7474*  7475*  7477*  7731*  7733*  7734   7737*  7738   7749*  7752*
                7753*  7755*  7757   7760   7762*  7784*  7785   7788   7818   7827*  7829*  7835*  7836*
                7855*  7857   7859   7861   7866   7868   7870   7872   7877*  7878   7996   7911   7915
                7944*  7945   7965   7982   7985   8003   8006   8038*  8039*  8040*  8082*  8083*  8084*
                8095*  8101*  8102*  8103   8110*  8112   8115   8123*  8124*  8125*  8127*  8148   8155*
                8156   8173*  8185*  8196*  8198*  8199*  8200*  8204   8249*  8251   8253   8255*  8260*
                8261*  8263   8269*  8272*  8283*  8286*  8291   8294   8297   8301   8303   8309   8318
                8324   8326*  8327*  8328*  8330   8337   8362*  8364   8369*  8370*  8371*  8372*  8373*
                8376*  8379   8382*  8385   8392   8396   8398*  8399*  8400   8408   8411   8421*  8422*
                8423*  8424   8425   8428*  8429*  8430*  8432   8434*  8437   8446*  8459*  8467*  8469
                8487   8493   8495   8498   8504*  8505*  8506*  8510   8513   8517*  8518*  8521*  8522
                8529   8531   8533   8535   8543   9545   8553*  8554*  8562*  8695   8774*  8775   8779*
                8780   8792*  8793*  8794*  8795*  8807   8809*  8810*  8811*  8812*  8813   8815*  8816*
                8829   8831*  8832   8833*  8843*  8844*  8845   8846*  8847   8849*  8850*  8873   8897*
R2      =%000002  3693*  4165*  4180   4182*  4184*  4187   4203*  4958   4960*  5026*  5028*  5044*  5046
                5065*  5317*  5325*  5461*  5466*  5481*  5487*  5496*  5504*  5509*  5521   5522*  5530*
                5534*  5541*  5560*  5568*  5576*  5593*  5596*  5834   5837*  5856*  5863*  6298   6314*
                6315*  6321*  6357*  6364*  6371*  6377*  6383   6395   6399*  6400*  6402*  6492   6525*
                6703   6717*  6718   6755*  6765   6767   6769   6771   6781   6782*  6789*  6902   6905*
                6912*  6935*  6974   6975*  6980   6982*  6998*  7140*  7148*  7150*  7152*  7471*  7472*
                7473*  7475*  7477*  7732*  7734   7736*  7738   7853*  7854*  7855   7875   7882*  7898*
                7900*  7913   7918*  7923   7962   7963   7979   7986   7990   7995   8007   8010   8017
                8027   8051   8058   8060   8061   8077*  8092   8094*  8114*  8118   8120*  8152   8158
                8201*  8203*  8215   8221*  8232   8234   8238*  8239*  8240*  8241*  8242*  8243*  8244*
                8250*  8256*  8339*  8352   8354*  8365*  8375*  8404*  8439   8442*  8497*  8501*  8512*
                8516*  8557   8559*  8564   8566*  8664   8667   8668*  8676*  8680*  8768*  8769*  8770*
                8771*  8772*  8775*  8780*  8811   8828*  8832*  8845*  8874   8896*
R3      =%000003  3693*  3963*  3968*  3974   4081*  4109*  4132*  4137*  4164*  4172*  5057*  5067*  5192*
                5195*  5414   5417*  5419*  5422*  5424*  5426*  5493*  5484*  5485   5486   5498*  5499*
                5500*  5501   5574*  5578*  5580*  6298   6300*  6306*  6319*  6321*  6492   6525*  6532*
                6537*  6568*  6596*  6604*  6613*  6631*  6635*  6650*  6664*  6703   6708*  6712*  6715*
                6747*  6755*  6866   6870*  6873*  6876*  6902   6906*  6911   6912*  6934*  6953*  6956*
                6962*  6974   6976*  6981   6982*  6939*  7045   7054*  7060*  7061*  7064*  7069*  7070*
                7071   7080*  7409   7411*  7412   7415*  7416   7423*  7424   7426   7434   7436   7442
                7444   7446*  7452*  7470*  7471*  7472*  7473*  7475*  7477*  7744*  7745*  7746*  7785*
                7786*  7787*  7800*  7833*  7945*  7946*  7947*  7963*  7966*  7968   7969   7970   7974
                7990*  7991*  7993*  7994   8007*  8008   8015   8022   8032   8034   8036   8041   8046
                8056   8058*  8065*  8070   8072   8087   8111*  8122*  8126*  8158*  8159*  8160*  8161*
                8166*  8168*  8169   8171   8270*  8281   8282*  8284   8287*  8460*  8461   8463*  8468
                8507*  8519*  8520   8547*  8555*  8561*  8670*  8672   8679   8681   8682   8773*  8776*
                8778*  8781*  8875   8895*
R4      =%000004  3693*  3964*  3965   3972   4080*  4086   4093*  4094   4096   4098   4105*  4112*  4166*
                4171*  4176   4186*  5056   5186*  5188*  5189   5197*  5414   5415*  5416*  5418*  5421*
                5423*  5426*  5524*  5525*  5526*  5527   5570*  5571*  5573   5591   6298   6299*  6302
```

```
                        6304*  6310   6318*  6321*  6344   6395   6397*  6398*  6402*  6492   6525*  6531*  6535*
                        6538   6548   6552   6554   6557*  6558*  6559   6567*  6570*  6576*  6577   6579   6581
                        6588*  6593*  6597*  6598   6600*  6601*  6602*  6603*  6606*  6618*  6621*  6622   6624*
                        6625   6632   6634*  6649*  6651   6654   6667*  6688   6798   6809*  6810*  6811*  6812*
                        6813*  6867   6868*  6869*  6872*  6875*  6880*  6881*  6882*  6883*  6884*  6902   6904*
                        6912*  6924   6949   6960   6974   6977*  6982*  7003   7046   7048*  7049*  7050*  7051
                        7052*  7066   7068*  7076*  7079*  7470*  7472*  7475*  7477*  7747*  7748*  7788*  7793*
                        7790   7794   7818*  7834*  7856*  7863*  7872*  7892   7920*  7921   7940   7964*  7965*
                        7967*  7968*  7969*  7984*  7985*  8005*  8006*  8109*  8132*  8147*  8148*  8184*  8201*
                        8219*  8224*  8225*  8258*  8266*  8270   8318*  8319*  8320*  8330*  8364*  8378*  8392*
                        8393*  8408*  8411*  8412   8413   8414   8415   8424*  8425*  8426   8486*  8487*  8503*
                        8528   8666*  8667*  8674   8678   8693   8694*  8695*  8696   8699*  8700   8704*  8705*
                        8876   8894*
R5      =%000005        3693*  3901*  3902*  3903   3913*  3914   3915   3916   3917   3920*  3921   3922   3923
                        3953*  3954   3956   3972*  3973   3979*  3984*  4133*  4134   4136*  4143   4150   4157*
                        4167*  4168   4170*  4184   4185*  4235*  4236   4238*  4241*  4242   4244*  4248*  4250
                        4252*  4254*  4269*  4270   4272*  4275*  4276   4278*  4302*  4303   4305*  4336   5056*
                        5058   5080   5125   5127*  5132*  5160   5189*  5193   5414   5426*  5523*  5524   5526
                        5529*  5535*  5537*  5538*  5539   5540   5573*  5575   5581*  5621*  5623   5627*  5628*
                        5629*  5630   5634*  5635*  5637*  5639*  5640   5644*  5645*  5647*  5649*  5650   5654*
                        5655*  5657*  5659*  5660   5666*  5667*  5669   5673   5674   5697   5703*  5708*  5710
                        5717*  5720*  5721   5726*  5728   5732*  5737   5749*  5750   5762*  5814   5912   5913*
                        5914*  5916   5917*  5918   5919   5920   5935*  5937*  5938   5940   6327*  6329*  6333*
                        6335*  6337*  6492   6493*  6495   6497   6499   6501*  6502   6506   6510   6514   6520
                        6525*  6533   6535   6594   6597   6608   6619   6621   6685   6694*  6695   6696   6697
                        6698*  6721*  6729*  6739*  6788*  6790*  6891   6893*  6902   6903*  6912*  6925   6946
                        6958   6974   6978*  6982*  7001   7047   7053*  7055*  7057*  7058*  7059*  7060   7078*
                        7093   7096*  7121   7470*  7471*  7472*  7475*  7477*  7799*  7800   7801   7803   7816*
                        7824*  7825*  8061*  8066   8314*  8321*  8334*  8344*  8350*  8359*  8367   8390*  8528*
                        8529   8671*  8672   8877   8893*
R6      =%000006        3693*  3893*
R7      =%000007        3693*
S       047521          5965   5976   9496*
SAVEFG  030564          4078*  7631*  8078   8210   8405
SAVER1  011322          5022*  5079*  5107*  5118   5126   5130*
SAVER5  011324          5023*  5080*  5108*  5127   5131*
SAVLOD  001304          3832*  3986   9622
SAVREG= 104416          7475   7477   7479*
SAVR05  037014          8872*
SAVR15  037014          7727   7783   7852   7909   8108   8183   8458   8485   8663   8767   8873*
SC      033626          8189   8249*  8262
SCMND   022466          6512   6618*
SC1     033666          8260*  8278
SC1A    033706          8259   8266*
SC10    034510          8325   8390*
SC11    034536          8313   8338   8396*
SC12    034626          8307   8316   8391   8411*
SC13    034676          8293   8296   8421*
SC2     033722          8218   8220   8226   8252   8254   8269*
SC3     033774          8286*  8323   8374   8381   8383   8395   8410   8448
SC4     033770          8284*  8288
SC5     034006          8285   8291*
SC6     034156          8302   8324*
SC6A    034274          8315   8350*
```

```
SC7     034426          8340    8355    8366    8376#
SC8     034464          8333    8343    8349    8358    8384#   8420
SDETAL  020556          6316    6343#   6401
SEARCH= 000131          3711#
SECOND  021624          3909*   5882    6432    6442*   6443    6445*   6446*   6447    6449*   6483#
SEEK  = 000105          3703#
SEEKFG  030566          7639#   7732    7929
SELDRV= 000145          3714#
SELPAR  014330          4227    4311    5618#
SEPART  023472          6720*   6767    6793#
SETFMT= 000143          3713#
SETVEC  002652          4048    4052    4056    4073#
SET.IE  036206          7759    7792    7894    7942    8133    8267    8693#
SHOTYP  020504          6309    6326#   6396
SIXTEE  021632          3905*   3906*   6438*   6440*   6441*   6488#
SIZMEM  001702          3927#
SKIER   010006          4511    4898#
SP    =%000006          3693#   3893*   3913    3920    3953    3973*   3979    3984    3996*   4010*   4017*   4021*   4025*
                        4029*   4033*   4037*   4041*   4045*   4057*   4062    4125    4229*   4231    4262*   4264    4285
                        4294*   4301    4325*   4326*   4328    4329    4532*   4533*   4535    4536    4543*   4545    4690*
                        4691*   4727*   4728*   4956*   4989*   4990*   4991*   4992    5125*   5132    5156*   5157*   5160*
                        5163*   5173*   5185*   5187    5199*   5200    5206*   5207*   5208*   5209*   5211    5213*   5214
                        5220*   5230*   5231*   5249*   5250*   5252    5262*   5263*   5265    5271*   5279*   5301*   5302*
                        5303*   5304*   5306    5308*   5309    5315*   5318*   5323*   5366    5367    5376*   5377    5390*
                        5407    5414*   5426    5457*   5470    5472*   5494*   5517    5521*   5532    5534    5602*   5607
                        5674*   5675*   5676*   5679*   5681*   5682    5737*   5762    5799    5820*   5821*   5823    5824
                        5834*   5840*   5841*   5843    5847    5849    5852    5862*   5863    5912*   5935    5940*   5955*
                        5959*   5960*   5968*   5972*   5973*   5977*   5978*   5993*   5996*   6001*   6004*   6005*   6010*
                        6013*   6021*   6024*   6032*   6037*   6038*   6043*   6044*   6055*   6059*   6067*   6070*   6073*
                        6074*   6075*   6080*   6083*   6090*   6093*   6095*   6096*   6097*   6098*   6100*   6113*   6120*
                        6124*   6153*   6154*   6168*   6169*   6173*   6176*   6177*   6181*   6182*   6195*   6196*   6200*
                        6201*   6205*   6208*   6239*   6240*   6241*   6243*   6244*   6245*   6246*   6248*   6250*   6261*
                        6280*   6289*   6298*   6321    6347*   6357*   6364*   6371*   6383*   6384*   6387*   6395*   6402
                        6437*   6468*   6476    6492*   6525    6546*   6572*   6685*   6688*   6689*   6690    6694    6698
                        6703*   6717    6755    6776*   6779    6780*   6781*   6783*   6784    6849*   6866*   6867*   6875
                        6876    6889*   6890*   6891*   6893    6894    6902*   6903    6905    6906    6910*   6911*   6912
                        6913*   6923*   6924*   6925*   6926*   6927*   6928*   6929    6931    6933*   6938    6940*   6941
                        6943*   6944    6948*   6950*   6951    6954*   6963*   6966*   6974*   6976    6978    6980*   6981*
                        6982    6995*   7004*   7006    7018*   7019    7029    7033    7034*   7045*   7046*   7047*   7053
                        7078    7079    7080    7081*   7091*   7107    7108*   7112    7113*   7116*   7131*   7133    7141*
                        7143    7145    7153*   7154    7156    7157*   7159    7165*   7166*   7173*   7175    7181*   7188*
                        7189*   7190*   7191*   7192*   7195*   7198*   7200    7206*   7207*   7222*   7223    7232    7233*
                        7262*   7263    7264*   7266    7267    7268*   7271    7273*   7275*   7281    7313*   7316*   7317
                        7318    7319*   7321    7322    7323*   7329    7355*   7356*   7361    7363    7369    7377    7394*
                        7395*   7398*   7399*   7409*   7410*   7415    7418    7422*   7429    7433*   7451    7452    7453*
                        7454*   7455*   7468    7469*   7470*   7471*   7472*   7473*   7474*   7475*   7476*   7477*   7478*
                        7479*   7728*   7764    7799    7805*   7809*   7816    7822*   7849*   7886    7910*   7937    7970*
                        7974*   7978*   7986*   7994*   7995*   7999*   8010*   8017*   8027*   8041*   8051*   8065    8072*
                        8067*   8096*   8152    8166    8208*   8277*   8280*   8281*   8284    8289    8299    8321    8334
                        8335*   8344    8346*   8359    8360*   8384    8390    8419    8463    8473*   8483*   8491    8540
                        8549    8581*   8587*   8588*   8589    8591    8596    8598    8600*   8601*   8604    8610*   8622
                        8623*   8631*   8632*   8633    8644    8651    8678*   8679*   8680    8693*   8696*   8697*   8698*
                        8702    8704    8705    8873*   8874*   8875*   8876*   8877*   8878*   8891*   8892    8893    8894
                        8895    8896    8897
```

```
SPOTCK  015560          4516    4614    4651    4697    4739    4780    4802    5834#
SRCHWT  030540          7585#   7911#   8003#   8400
STACK = 001100          3693#   3832    3893    3996
STACK1  001220          3832#   4125    4285
STAHD1  051572          6338    9558#
START   001430          3722    3723    3893#   6235    6292
STATHD  051457          6330    9556#
STATIN  001222          3832#   3899#   4128    4130#   6475#
STATIS  013262          4295    5431#
STATPR  020362          4131    6298#   6638
STKLMT= 177774          3693#
STNDAT  042556          5189    5197    5573    5581    9093#
ST0     035134          8465    8483#
ST01    035176          8493#   8536    8541
ST02    035374          8492    8494    8496    8528#
ST03    035444          8530    8542#
ST04    035470          8527    8546    8548#   8552    8558    6569
ST05    035502          8490    8509    8539    8551#
ST06    035510          8532    8553#
ST07    035546          8534    8561#
ST08    035576          8560    8565    8567#
SVPH11  036110          8080    8095    8121    8212    8223    8306    8345    8363    8407    8443    8502    8568    8663#
SWR   = 177570          3693#   4318    4320    4359    4418    5008    5062    5153    5298    5619    5746    5797    5874
                        5926    6018    6052    6187    6212    6516    6826    6853    7102    7123    7468
SWTIM   004730          4381    4425#
SW0   = 000001          3693#   4318    5619    6516
SW00  = 000001          3693#
SW01  = 000002          3693#   5008
SW02  = 000004          3693#   4359
SW03  = 000010          3693#
SW04  = 000020          3693#   5797    6826    6853
SW05  = 000040          3693#   5926    6018
SW06  = 000100          3693#   5746
SW07  = 000200          3693#   5153
SW08  = 000400          3693#
SW09  = 001000          3693#
SW1   = 000002          3693#
SW10  = 002000          3693#   5874
SW11  = 004000          3693#
SW12  = 010000          3693#
SW13  = 020000          3693#   7102    7123
SW14  = 040000          3693#
SW15  = 100000          3693#   4418    6187    6212
SW2   = 000004          3693#
SW3   = 000010          3693#   5062    5298
SW4   = 000020          3693#
SW5   = 000040          3693#   6052
SW6   = 000100          3693#
SW7   = 000200          3693#   4320
SW8   = 000400          3693#
SW9   = 001000          3693#
SYSTAT  051437          4083    9555#
T       047475          5958    5971    9494#
TBITVE= 000014          3693#
```

```
TD      033344          8187    8196#   8300
TIMER   030570          7644#   7833#   7947#   8083#   8122#   8199#   8327#   8372#   8422#   8429#   8461    8463#   8519#
                        8547#   8555#   8561#
TITLE   052010          3894    9566#
TKVEC = 000060          3693#   7346#   7347#
TPVEC = 000064          3693#
TRAPVE= 000034          3693#   3893#   3897#
TRFER   002706          4473    4878#
TRNSWT  030536          7577#   7962#   8105#   8114    8130#   8201    8202#   8497    8512    8525#
TRTVEC= 000014          3693#
TYPOS = 104410          3976    6850    7469    7479#
TYPE  = 104400          3894    3911    3918    3951    3962    3970    3977    3982    4063    4064    4066    4082    4083
                        4084    4085    4106    4108    4114    4116    4155    4177    4178    4179    4390    4391    4392
                        4393    4394    4406    4407    4412    4413    4414    4415    4416    4438    4439    4440    4441
                        4442    5876    6231    6232    6233    6290    6345    6346    6350    6357    6364    6371    6374
                        6375    6376    6383    6390    6429    6430    6431    6432    6494    6524    6641    6642    6645
                        6646    6647    6655    6656    6659    6661    6663    6669    6686    6691    6692    6704    6705
                        6706    6707    6753    6799    6800    6801    6802    6804    6814    6815    6816    6817    6820
                        6821    6847    6848    6857    6858    6859    6860    6862    6863    6864    6879    7031    7161
                        7163    7202    7204    7312    7360    7362    7367    7421    7427    7432    7438    7440    7443
                        7447    7449    7468    7469    7470    7471    7479#
TYPEST  021110          4156    6395#   6626    6865
TYPOC = 104402          3981    7469    7479#
TYPON = 104406          7479#
TYPOS = 104404          7479#
TYPRI4  025046          6327    6329    6333    6335    6337    7091#
UCPAR   004652          4377    4406#
ULDFLG  030544          7604#   7859    7877#   8040#   8101#   8124#   8303    8379    8382#   8396    8398#   8505#   9518#
UNIT    001254          3832#   4079#   4084    4111#   4175#   4176#   4178    4372#   4373#   4393    4415    4441    5915#
                        5916#   5923    6343#   6344#   6345    6653#   6654#   6655    6687#   6690#   6691    6705    6797#
                        6798#   6801    6818#   6819#   6820    6851#   6852#   6859    6863
UNLOAD= 000103          3702#
UNSAF   010142          4467    4928#
UNTASN  051347          6536    9551#
UNTMSG  051277          4177    6800    9547#
UNTNOT  051325          6610    6628    9550#
UNTOFF  051304          4091    6574    9548#
UNTON   051315          4089    9549#
WAIT    041646          4147    4209    4250    4254#   4261    4279    9004#
WC      033520          8213    8230#
WCFER   010044          4491    4909#
WCKD  = 000151          3715#   9018
WCKER   006060          4470    4598#
WCKHD = 000153          3716#   9019
WLEER   007520          4497    4838#
WRTDAT= 000161          3717#   5812    9020
WRTHD = 000163          3718#   9021
WRTPK   015252          4218    5775#
WRTPK1  015302          5777    5779    5763#
WRTPK2  015452          5782    5806#
WRTPK3  015476          5781    5805    5810#
WRTPK4  015530          5809    5815#
WRT.AD  036036          8624#   8637#   8640#   9465
WRT.RP  035746          7806    7810    7971    7975    7979    7987    7996    8000    8011    8018    8029    8042    8052
```

```
                    8073    8088    8097    8347    8622#
WRT.WD  036034      8622*   8626    8633*   9639#   9465
ZROIND  011320      5021*   5072*   5078*   5082    5103#
$BOAOR  001122      3832#
$BODAT  001126      3832#
$BOSEC= 000114      5835    6709    8945#   8946
$BELL   001152      3832#   5876    7367    7468
$BUF  = 000006      4231*   4264*   4545*   4991    5013    5432    5500    5506    5511    5519    5548    5559    6067
                    6074    8912#   8913
$CMTAG  001100      3832#   3893
$CM3  = 000000      3832#
$CNTLC  026475      7449    7460#
$CNTLU  026470      7438    7459#
$CODE = 000030      4191*   4200    5005    5030    5088    5215    5312    5437    5441    5556    5561    5584    5712
                    5714    5738    5748*   5758    5783    5807*   5811*   5890    8232    8234    8239    8242*   8922#
                    8923
$COMMD= 000002      4193*   5335*   5346*   5357*   5370*   5750*   5808*   5812*   8243*   8244*   8909#   8910
$CRLF   001157      3832#   3895    3962    4082    4108    4114    4390    4394    4406    4412    4416    4432    4442
                    4947    4956    5145    5165    5176    5274    5282    5327    5329    5877    5887    5898    5910
                    5921    5944    5981    5987    6007    6026    6047    6061    6077    6103    6114    6126    6132
                    6142    6157    6185    6186    6210    6211    6231    6233    6328    6334    6374    6390    6429
                    6494    6663    6669    6804    6814    6815    6821    6847    6860    6864    6879    7360    7468
                    7469
$CYL  = 000012      4190*   4537*   5016    5369*   5447    5563    5610    5741    5753*   5787*   5788    5795*   6032
                    8240    8915#   8916
$DBLK   027444      7471#
$DB2D   027774      6170    6178    6183    6197    6202    6357    6364    6371    7475#   7476
$DB2O   030220      7477#   7478
$DECVL  030154      7475#
$DRVID= 000154      6648    6695*   6696*   6697*   8946#   8947
$DSPLY  025074      7102#   7480
$DTBL   027434      7471#
$ENDAD= ****** U    3832
$ERFLG  001103      3832#   7468*
$ERMAX  001115      3832#
$ERROR  026514      3893    7468#
$ERRPC  001116      3832#   7468*   7469
$ERRTB  001340      3832#   7469
$ERRTY  026626      7468    7469#
$ERTTL  001112      3832#   7468*
$FILLC  001150      3832#   7271
$FILLS  001147      3832#
$FMT  = 000001      4560*   8908#   8909
$GOAOR  001120      3832#
$GODAT  001124      3832#
$HARD = 000066      6387    6420*   8933#   8934
$HD   = 000000      3678
$HIMUM  027674      4000*   5587*   5591    5595    6890    7473#*
$ICNT   001104      3832#
$ITEMB  001114      3832#   7468*   7469
$LF     001160      3832#   7447    7468
$LINAD  016322      5918*   5919*   5920*   5929*   5930*   5933*   5937    5947#
$LKCSB  001170      3832#   6277*
$LKCSR  001166      3832#   6271    6278*
```

DZRPNB.P11 CROSS REFERENCE TABLE

```
$LKS    001174          3833*  6282   6287*
$LLVEC  001176          3832#  6284
$LONUM  027676          3999*  5586*  5590   5592   5627   6889   7473#*
$LPADR  001106          3832#
$LPERR  001110          3832#
$LPVEC  001172          3832#  6274
$LSCS   001206          3832#  6258   7307   7327
$LSDB   001210          3832#  7329*
$LSTAD  027772          3928   3933   7474#*
$MASK = 000024          4524*  4552   4602*  4663*  4710*  4752*  4788*  4816*  4887*  4917*  5394   5399   8920#
                        8921
$MISPO= 000072          6205   6387   6422*  8935#  8936
$NCODE= 000102          4334*  5630*  5661   5677   5686   5714*  5715*  5721*  5739   5748   5749   8938#  8939
$NCYL = 000106          4324*  5660*  5753   8942#  8943
$NEXT = 000112          4214   4337*  5698*  5761*  5815*  8944#  8945
$NPATC= 000103          4336*  5697*  5751   8939#  8940
$NSEC = 000104          4329*  5640*  5752   8940#  8941
$NTRK = 000105          4328*  5650*  8941#  8942
$NULL   001146          3832#  7273
$NWRDL= 000110          4330*  4333*  5673*  5692*  5695*  5754   5755   8943#  8944
$OCNT   027224          7470#*
$OCTVL  030322          7477#
$OMODE  027226          7470#*
$OPERC= 000042          5608*  5609*  5776   5778   6168   6195   6315   6400   6869   8927#  8928
$PACK = 000031          4216   4239   4273   5780   5804   6566*  8923#  8924
$PASS   001100          3832#
$PASSC= 000074          6547   6563*  6849   6855   6877*  8936#  8937
$PATTC= 000034          4194*  4195*  5571   5751*  5814*  8925#  8926
$POSIT= 000046          5449*  5450*  5612*  5613*  6200   6842   6845   8928#  8929
$PREVA= 000036          5610   5741*  5742*  5745*  5785*  5823*  5824*  5968   5973   5978   5993   8240*  8241*
                        8926#  8927
$PREVO= 000032          5738*  5783*  5894   8239*  8924#  8925
$PRFLG  001204          3832#  6255*  6259*  7105   7303
$PRINT  025656          7109   7303#
$PRTER  001212          3832#  7309   7311*  7315*
$PSEL = 000003          8910#  8911
$QUES   001156          3832#  6707   7163   7204   7440   7468
$RAND   027550          5589   5618   5671   5684   5730   5775   7473#
$RDCHR  026200          7394#  7479
$RDDEC= ****** U        7479
$RDLIN  026234          7409#  7479
$RDOCT= ****** U        7479
$READ = 000056          5445*  5446*  6181   6836   6839   8930#  8931
$REG  = 000014          5422   8916#  8917
$RESRE  027512          7472#  7479
$RETRY= 000022          4529*  4556*  4557*  4562*  4607*  4622*  4627*  4630*  4631*  4664*  4711*  4753*  4789*
                        4817*  4886*  4916*  5389*  5401*  5402*  6154   8919#  8920
$RHAS = 000202          8954#  8955   9475
$RHBA = 000170          4992   5205   5236   5257   5300   5320   5431   6021   6073   6089*  6090   8949#  8950
                        9475
$RHCA = 000220          5447   5787   6059   8961#  8962   9479
$RHCC = 000222          4324   4537   4734   5369   5745   5785   5838   5955   5996   8962#  8963   9479
$RHCS1= 000164          4351   4353   4966   8947#  8948   9470
$RHCS2= 000174          4468   4658   4704   4746   4852   4867   4884   4968   8951#  8952   9470
```

```
$RHDA = 000172          5786    6005    6055    6240    6245    6250    8950#   8951    9475
$RHDB = 000206          6093    8956#   8957    9475
$RHOSI= 000176          4355    4471    4500    8952#   8953    9470
$RHDT = 000212          8958#   8959    9475
$RHEC1= 000230          5221    6120    8965#   8966    9470
$RHEC2= 000232          5239    5422    6124    6305    6312    6317    6666    8966#   9470
$RHER1= 000200          4474    4477    4480    4483    4486    4489    4492    4495    4498    4503    4506    4525    4552
                        4554    4599    4620    4633    4635    4724    4970    5396    5399    8953#   8954    9470
$RHER2= 000224          4972    8%3#    8964    9470
$RHER3= 000226          4509    4974    8964#   8965    9470
$RHLA = 000204          8955#   8956    9475
$RHMR = 000210          8957#   8958    9475
$RHOF = 000216          8960#   8961    9479
$RHSN = 000214          8959#   8960    9479
$RHWC = 000166          4987    5015    5206    5302    6024    6095    8948#   8949    9475
$RPADR  001162          3832#   4074    9618
$RPVEC  001164          3832#   4075    9620
$RPZR0  024560          6364    6371    7017#
$RPZR4  024540          6383    6389    7013#
$RPZR5  024550          6349    7015#
$RPZR8  024530          6357    7011#
$RSAV = 000076          5586    5587    5590*   5591*   8937#   8938
$SAVRE  027454          7472#   7479
$SB20   030170          5174    5232    5956    5962    5964    5969    5974    5979    6011    6033    6039    6045    6348
                        6383    6388    7476#
$SB20   030340          6081    7478#
$SEC  = 000010          4188*   4536*   5018    5367*   5565    5742    5752*   5786*   5794*   6044    8241    8913#   8914
$SETUP= 000006          3883#   3893
$SIZE   027700          3927    7474#
$SKI  = 000070          6208    6387    6421*   8934#   8935
$SOFT = 000064          6387    6419*   8932#   8933
$SSEC = 000026          4199*   4202*   4538    4540    5024    5028    5029    5209    5304    5757*   5760*   5806*   5810*
                        6098    8921#   8922
$STUP = 177777          3883#
$SUPRS  025510          5175    5233    5957    5963    5966    5970    5975    5980    6012    6034    6040    6046    6082
                        6171    6179    6184    6198    6203    7222#
$SWR  = 122000          3667#   3678    3679    3832    3893    7468
STATUS= 000016          4291    4349    4366    4374    4376    4378    4380    4382    4384    4465    4547    4550    4564
                        4624    4687    5350    5374    5386    5392    5896    8917#   8918    9479
$TERM = 000024          7479    7481#
$TIME   021314          6326    6427#   7125    7359
$TKB    001140          3832#   7348    7355    7398
$TKCNT  026172          7344*   7357    7370*   7374*   7381#
$TKINT  026000          4115    7344#
$TKQEN= 026200          7365    7384#
$TKQIN  026174          7345*   7365    7369*   7371*   7375*   7382#
$TKQSR  026176          6493    7345    7375    7376*   7383#
$TKS    001136          3832#   3893    7349*   7396
$TKSRV  026042          7346    7355#
$TN   = 000001          3668#   3678
$TNPWR  030104          7475#
$TOTAL= 000062          6173    6384    6423*   6828    8931#   8932
$TPB    001144          3832#   7281*
$TPFLG  001151          3832#   3900*   7258
```

DZRPNB.P11 CROSS REFERENCE TABLE

```
$TPS    001142          3832#   7279
$TRANS= 000052          5435*   5436*   5443*   5444*   6176    8929#   8930
$TRAP   030370          3893    7479#
$TRK  = 000011          4189*   4535*   5366*   5791    5793*   6038    8914#   8915
$TRP  = 000024          7479#   7480#
$TRPAD  030422          7479#   7481
$TSTNM  001102          3832#   7468
$TTYIN  026502          7411    7412    7424    7450*   7455    7461#
$TYPBN= ****** U        7479
$TYPOS  027230          7471#   7479
$TYPE   025550          7094    7114    7258#   7479
$TYPOC  027026          7470#   7479
$TYPON  027042          7470#   7479
$TYPOS  027002          7470#   7479
$WRDL = 000020          4196*   4538    4540*   4541    4989    5014    5207    5301    5443    5464    5473    5475*   5476
                        5477*   5498    5507    5512    5520    5552    5560    5754*   5801#   6070    6096    8918#   8919
$WRDM = 000004          4197*   4196*   4541*   4542*   5755*   5756*   5802*   5803*   8911#   8912
$STRP = 000002          7479#   7480
$OFILL  027225          7470#*
.     = 053564          3722#   3832#   3893    3904    4126    4129    4237    4271    4286    4419    4466    4469    4472
                        4475    4478    4481    4484    4487    4490    4493    4496    4504    4507    4510    4659    4688
                        4695    4705    4725    4735    4747    4853    4868    5006    5009    5025    5027    5033    5036
                        5039    5042    5237    5420    5425    5459    5515    5549    5551    5553    5555    5678    5875
                        6188    6213    6486#   6730    6740    6952    7383#   7384    7461#   7468    7469#   7471#   7475#
                        7477#   7479    7481    8292    8295    8312    8353    8380    8752#   8753#   8754#   8755#   8756#
                        8757#   8758#   8759#   8760    8980    8986#   9056#   9459#   9487#   9586#   9666#   9672
```

```
A       6404#   6419    6420    6421    6422    6423
COMMEN   904#   3693#
ENDCOM   916#   3693#
ERROR   3693#   8265    8322    8394    8593    8650
ESCAPE  1018#   3693#
MORETA  3725#   3832
MULT    3201#   3693#
NEWTST   956#   3693#
POP     1351#   3693#   5132    5426    5863    6321    6402    6525    6755    6912    6982    7471    7472    7473
PUSH    1343#   3693#   5125    5414    5834    6298    6395    6492    6703    6902    6974    7471    7472    7473
SCOPE   3693#
SETTRA  7479#   7480
SETUP    738#   3693#   3893
SKIP    1052#   3693#
SLASH    856#   3693#
SPACE   3693#
STARS    825#   3693#   3695    3699    3832    3884    3891    4119    4123    5866    5870    6223    6227    7211    7236
        7284    7332    7387    7402    7464    7468    7469    7470    7471    7472    7473    7474    7475    7476    7477
        7478    7479    7483    7493    8900    8904    8988    8992    9087    9091    9372    9376    9660    9664    9668
TRMTRP  7479#   7481
TYPBIN  1287#   3693#
TYPDEC  1257#   3693#   7469
TYPNUM  1224#   3693#
TYPOCS  1177#   3693#
TYPOCT  1140#   3693#   7469
TYPTXT  1094#   3693#
$SCMRE  3832#
$SCMTM  3832#
$SESCA  1031#   3693#
$SNEWT   985#   3693#
$SSET   7479#   7480
$SSKIP  1065#   3693#
.EQUAT   164#   3670#   3693
.HEADE    39#   3670#   3678
.K..1    300#
.SETUP   684#   3670#   3883
.SWRHI    81#   3670#   3679
.SWRLO  3670#   3679#   3680    3681    3682    3683    3684    3685    3686    3687    3688    3689    3690    3691    3692
.$CATC   478#   3671#   3722
.$CMTA   535#   3671#   3832
.$D82D  3401#   3674#   7475
.$D82O  3525#   3674#   7477
.$DIV   3303#
.$EOP   1399#
.$ERRO  1773#   3671#   7468
.$ERRT  1928#   3671#   7469
.$MULT  3239#
.$POWE  2964#
.$RAND  3021#   3673#   7473
.$RDDE  2660#
.$RDOC  2568#
.$READ  2354#
.$SAVE  2736#   3673#   7472
.$SB2D  3486#   3673#   7476
```

```
 $SB20   3588#   3674#   7478
.$SCOP   1575#
.$SIZE   3083#   3674#   7474
.$SUFR   3626#
.$TRAP   2837#   3673#   7479
 $TYPB   2269#
.$TYPD   2191#   3672#   7471
.$TYPE   2016#
.$TYPO   2094#   3672#   7470
```

ADC 5436 5444 5446 5450 5609 5613 6930 6942 6943 6947 6948 6950 6959 7002 7473
7475
ADC 3990 3992 4136 4145 4152 4170 4182 4202 4203 4252 4957 4958 4991 5015 5197
5199 5207 5217 5220 5235 5256 5302 5315 5390 5416 5421 5435 5443 5445 5449
5468 5476 5484 5500 5503 5520 5526 5529 5535 5538 5608 5612 5629 5639 5649
5659 5679 5760 5769 5836 5853 5862 5904 6096 6169 6177 6182 6196 6201 6261
6280 6289 6305 6312 6315 6317 6357 6364 6371 6377 6400 6572 6666 6710 6749
6779 6784 6788 6869 6929 6931 6941 6944 6946 6949 6958 6960 6963 6966 7001
7003 7030 7050 7116 7154 7166 7192 7195 7207 7268 7313 7323 7469 7470 7471
7473 7475 7477 7966 7967 7993 8068 8126 8168 8583 8637 8679 8812 8846
ASL 4093 4990 5213 5244 5308 5475 5499 5525 5732 5901 5902 5903 5917 6138 6557
6576 6601 6624 6775 6777 6778 6812 6883 7147 7149 7151 7188 7190 7191 7469
7473 7786 7946 8082 8198 8326 8371 8421 8428 8793 8810 8831 8844
ASLB 4195 7471 8256 8287
ASR 4105 5226 5227 5228 5231 5434 5477 6075 6567 6588 6603 6689 7477 8084 8160
8161 8200 8328 8373 8423 8430 8795 8816 8833 8850
BCC 7000 7471 8257
BCS 6932 6957
BEQ 3955 3957 3966 4048 4052 4087 4095 4097 4099 4110 4126 4129 4142 4144 4149
4151 4181 4201 4212 4217 4222 4224 4232 4251 4286 4290 4292 4309 4323 4340
4352 4367 4413 4466 4469 4475 4478 4481 4484 4487 4490 4493 4496 4499 4501
4504 4510 4548 4551 4553 4555 4561 4565 4622 4625 4632 4634 4659 4688 4705
4725 4735 4747 4853 4868 4885 4975 4993 5009 5033 5036 5039 5042 5061 5063
5066 5071 5083 5150 5171 5190 5212 5242 5254 5273 5299 5307 5321 5351 5375
5387 5393 5400 5433 5440 5448 5462 5467 5474 5480 5482 5497 5502 5528 5557
5562 5572 5577 5594 5597 5611 5636 5646 5656 5662 5678 5707 5713 5759 5768
5875 5897 5909 5939 6019 6053 6188 6213 6332 6373 6461 6471 6496 6534 6555
6565 6569 6578 6580 6582 6595 6599 6605 6609 6620 6623 6640 6644 6652 6665
6719 6738 6748 6766 6768 6774 6829 6835 6939 6952 7067 7136 7139 7142 7178
7180 7183 7225 7227 7364 7430 7435 7468 7469 7470 7477 7758 7791 7795 7802
7914 7922 8033 8035 8067 8071 8093 8113 8116 9119 8216 8231 8233 8292 8295
8325 8353 8386 8433 8440 8499 8511 8514 8546 8558 8565 8597 8645 8665 8669
8808 8830
BGE 7928 7992 8167 8172 8252 8608
BGT 5624 6955 7074 7144 7185 7470 7471 7477 7479 7858 7916 8157 8273 8302 8338
8438 8464 8470 8523
BHI 4007 5025 5047 5237 5258 5442 5711 5790 6726 6735 6745 6837 6843
BHIS 4013 4016 4020 4024 4028 4032 4036 4040 4044 4332 4539 5459 5515 5549 5553
5683 5694 6419 6420 6421 6422 6423 6498 6772 6840
BIC 4002 4733 5224 5225 5250 5251 5263 5264 5628 5720 7064 7153 7356 7399 7470
7477 7753 7825 8127 8632
BICB 4079 4175 4372 5715 5915 6343 6501 6653 6687 6783 6797 6818 6851 7911 8242
8243
BIS 4737 4771 5017 5252 5265 5564 5717 7069 7070 7470 7471 7475 7882 7898 7900
7918 8077 8120 8203 8221 8339 8354 8365 8375 8404 8442 8501 8516 8559 8566
8600 8697
BISB 4176 4373 5708 5916 6344 6654 6690 6798 6819 6852 7469 8003 8244
BIT 4318 4320 4351 4353 4355 4359 4366 4374 4376 4378 4380 4382 4384 4418 4465
4468 4471 4474 4477 4480 4483 4486 4489 4492 4495 4498 4500 4503 4509 4525
4550 4552 4554 4599 4620 4633 4635 4658 4687 4694 4704 4724 4731 4746 4852
4867 4884 4966 4968 5008 5062 5153 5298 5392 5399 5619 5739 5746 5758 5797
5874 5896 5926 6018 6052 6187 6212 6516 6826 6853 7102 7123 7468 7790 7794
7992 7921 7940 8284 8311 8426 8589 8596 8644 8700
BITB 4200 5005 5437 5556 5561 5584 5677 5712 8400 8529

DZRPN8.P11 CROSS REFERENCE TABLE

```
BLE     5045  5087  5569  7869  8304  8310  8397  8693
BLO     3929  3946  5092  5096  5789  6500  6770  6838  6644  6846  6856  7425
BLOS    4009  5465  5792  7413  7735  7739
BLT     7075  7146  7187  7276  7470  7471  7475  7897  7924  8170  8186  8262  8462  8627
BMI     4088  4350  4626  5077  5121  5136  5388  5781  5805  5855  6556  6724  6733  6743  7106
        7471  7823  7930  8209  8336  8351  8368  8492  8595  8675
BNE     3893  3904  3938  3943  3975  3987  3989  4054  4056  4135  4138  4169  4173  4210  4215
        4237  4240  4243  4265  4271  4274  4277  4304  4319  4321  4342  4354  4356  4360  4375
        4377  4379  4381  4383  4385  4472  4526  4558  4586  4600  4636  4695  4732  4939  4967
        4969  4971  4973  4988  5006  5031  5059  5068  5089  5119  5137  5152  5154  5194  5196
        5216  5313  5326  5395  5397  5403  5420  5425  5438  5488  5505  5531  5542  5579  5585
        5620  5670  5687  5719  5729  5740  5747  5777  5779  5798  5839  5844  5850  5857  5860
        5879  5927  6303  6307  6311  6320  6419  6420  6421  6422  6423  6428  6439  6444  6448
        6453  6457  6463  6473  6503  6507  6511  6515  6517  6521  6539  6549  6633  6636  6658
        6713  6827  6854  6874  7005  7021  7026  7065  7103  7124  7265  7272  7310  7320  7358
        7366  7417  7419  7437  7445  7468  7469  7470  7471  7473  7475  7754  7761  7804  7826
        7860  7862  7867  7871  7876  7879  7893  7941  8009  8016  8023  8037  8047  8057  8104
        8128  8154  8235  8254  8264  8271  8285  8288  8298  8312  8380  8402  8427  8456  8494
        8496  8530  8532  8534  8536  8544  8590  8625  8673  8701  8777  8782  8814  8848
BPL     3967  4507  4549  4566  5846  6242  6247  6551  7063  7259  7280  7304  7308  7328  7397
        7468  7470  7471  7763  7817  8079  8211  8278  8361  8406  8541
BR      3940  3959  3969  3980  3985  3991  4090  4092  4101  4103  4113  4139  4146  4153  4160
        4174  4183  4204  4219  4225  4247  4253  4281  4408  4528  4578  4580  4609  4611  4613
        4629  4638  4640  4666  4713  4755  4791  4819  4824  4889  4891  4919  4924  4945  5027
        5048  5053  5055  5069  5075  5081  5085  5124  5143  5198  5218  5246  5260  5280  5310
        5328  5391  5398  5469  5478  5489  5510  5533  5543  5551  5555  5582  5588  5598  5626
        5672  5685  5688  5709  5716  5731  5743  5770  5782  5809  5848  5851  5858  5943  5953
        6249  6260  6279  6288  6308  6313  6466  6505  6509  6513  6519  6523  6571  6584  6586
        6607  6614  6627  6630  6637  6660  6668  6722  6730  6740  6741  6750  6754  6785  6787
        6841  6861  6878  6945  7012  7014  7016  7024  7056  7077  7111  7155  7158  7164  7197
        7199  7205  7261  7278  7306  7326  7368  7372  7428  7439  7441  7448  7469  7470  7471
        7474  7475  7477  7479  7768  7793  7828  7837  7865  7874  7881  7889  7891  7895  7899
        7901  7919  7926  7932  7933  7935  7943  7948  8004  8014  8021  8031  8055  8069  8091
        8188  8218  8220  8226  8237  8259  8315  8323  8340  8355  8366  8417  8420  8431  8445
        8466  8509  8527  8552  8560  8569  8599  8634  8677  8703
BYS     7193  7196
CLC     3935  6961
CLR     3893  3899  3902  3963  3997  3998  4080  4130  4157  4158  4165  4166  4167  4185  4229
        4248  4262  4298  4325  4326  4343  4530  4532  4533  4543  4602  4690  4691  4727  4729
        4887  4960  5012  5022  5023  5130  5131  5208  5240  5248  5259  5303  5418  5423  5566
        5567  5570  5675  5761  5815  5820  5821  5840  5841  5889  5893  5913  5959  5960  5972
        5977  6004  6037  6043  6097  6153  6239  6244  6257  6259  6270  6272  6273  6283  6299
        6397  6530  6531  6593  6618  6649  6728  6764  6780  6809  6872  6880  6904  6907  6908
        6933  6940  6975  6977  6993  6994  7017  7054  7139  7140  7181  7315  7344  7374  7376
        7410  7433  7469  7470  7471  7473  7475  7477  7733  7749  7756  7787  7854  8105  8110
        8111  8130  8202  8238  8249  8280  8283  8459  8460  8506  8507  8525  8670  8676  8769
        8770  8771  8772  8828
CLRB    3900  3910  4067  5011  5078  5093  5097  6600  6811  6882  7160  7201  7446  7471  7475
        7767  7829  7835  7885  8039  8101  8102  8123  8124  8173  8191  8196  8376  8382  8398
        8399  8434  8446  8472  8504  8505  8517  8518  8553  8554  8562  8792
CLZ     6964
CMP     3893  3903  3928  3945  4006  4008  4012  4015  4019  4023  4027  4031  4035  4039  4043
        4052  4053  4096  4098  4125  4143  4150  4221  4285  4331  4341  4538  4734  4992  5024
        5032  5035  5046  5058  5193  5236  5257  5320  5447  5458  5464  5501  5514  5527  5548
```

E01


```
        5552  5610  5623  5682  5693  5710  5788  5838  6419  6420  6421  6422  6423  6460  6579
        6581  6725  6734  6744  6828  6836  6839  6842  6845  6855  7363  7365  7412  7424  7471
        7475  7479  7734  7738  7801  7803  8066  8103  8115  8152  8169  8171  8297  8469  8498
        8513  8522  8535  8591  8672  8682  8813  8847
CMPB    3954  3956  5030  5088  5091  5095  5215  5312  5441  5661  5686  5791  5843  5849  6443
        6447  6452  6456  6462  6495  6497  6499  6502  6506  6510  6514  6520  6533  6594  6608
        6619  6718  6767  6769  6771  7020  7137  7143  7145  7179  7184  7186  7226  7271  7416
        7434  7436  7444  7875  7923  8008  8015  8022  8032  8034  8036  8046  8056  8070  8156
        8232  8234  8626  8807
COM     7311  7824
DEC     4109  4137  4172  5065  5067  5151  5195  5222  5243  5325  5419  5466  5479  5481  5487
        5504  5530  5536  5541  5576  5578  5593  5596  5856  6306  6319  6568  6604  6635  6664
        6712  6747  6873  6954  7004  7027  7228  7423  7469  7475  7477  7762  8261  8272  8606
        8776  8781
DECB    4557  4631  5402  6241  6246  6450  7062  7073  7275  7470  8843
EMT     3693
HALT    3722  4127  4287  4420  5338  5349  5360  5373  5550  5554  5606  6189  6214  6234  6291
        7260  7305  7468  7479
INC     3968  4112  4171  4559  5020  5049  5073  5098  5508  5509  5513  5637  5647  5657  5667
        6304  6318  6419  6420  6421  6422  6423  6438  6570  6606  6634  6667  6731  6782  6789
        6877  6953  7023  7068  7076  7370  7371  7468  7470  7471  7473  7475  7477  7752  8125
        8255  8286  8467  8521  8603
INCB    4111  4556  4627  4630  5090  5094  5389  5401  6442  6446  6451  6455  6459  6465  7468
        8155  8809
IOT     3693
JMP     3722  3723  3930  4011  4018  4022  4026  4030  4034  4038  4042  4046  4058  4068  4117
        4257  4266  4314  4368  4402  4450  4467  4470  4473  4476  4479  4482  4485  4488  4491
        4494  4497  4505  4508  4511  4512  4594  4689  4696  4726  4736  4738  4806  5099  5238
        5410  5800  5900  5990  6148  6163  6235  6292  6541  6544  6547  6675  6681  7168  7209
        7983  8293  8296  8300  8313  8374  8381  8383  8388  8389  8395  8410  8448
JSR     3927  3978  3983  4073  4076  4077  4115  4131  4156  4159  4218  4226  4227  4228  4230
        4233  4234  4246  4256  4263  4267  4268  4280  4295  4297  4299  4300  4306  4307  4311
        4313  4327  4335  4344  4357  4358  4361  4395  4397  4398  4399  4400  4401  4417  4425
        4427  4428  4429  4430  4431  4443  4445  4446  4447  4448  4449  4454  4458  4459  4460
        4516  4518  4520  4521  4522  4523  4527  4531  4534  4544  4546  4563  4567  4568  4569
        4570  4571  4572  4574  4575  4576  4577  4579  4581  4582  4584  4587  4589  4590  4592
        4598  4605  4606  4608  4610  4612  4614  4619  4623  4628  4637  4639  4641  4642  4644
        4645  4646  4651  4653  4655  4656  4657  4660  4661  4662  4665  4667  4668  4670  4671
        4672  4677  4679  4680  4681  4682  4692  4693  4697  4699  4701  4702  4703  4706  4707
        4708  4709  4712  4714  4715  4717  4718  4719  4729  4730  4739  4741  4743  4744  4745
        4748  4749  4750  4751  4754  4756  4757  4759  4760  4761  4766  4767  4769  4770  4772
        4773  4774  4775  4780  4782  4784  4785  4786  4787  4790  4792  4793  4795  4796  4797
        4802  4804  4810  4812  4813  4814  4815  4818  4820  4821  4823  4828  4830  4831  4832
        4833  4838  4840  4841  4842  4847  4849  4850  4851  4854  4855  4856  4857  4862  4864
        4865  4866  4869  4870  4871  4872  4873  4878  4880  4881  4982  4883  4888  4890  4892
        4893  4898  4900  4901  4902  4903  4904  4909  4911  4912  4913  4914  4915  4918  4920
        4921  4923  4928  4930  4931  4932  4933  4940  4942  4943  4944  4956  4976  4978  4979
        4980  4981  4982  4994  4996  4997  4998  4999  5000  5034  5037  5040  5043  5050  5052
        5054  5064  5074  5084  5100  5122  5123  5128  5129  5133  5138  5140  5141  5142  5158
        5161  5164  5174  5175  5177  5178  5210  5232  5233  5271  5279  5305  5319  5324  5336
        5347  5358  5371  5385  5405  5406  5589  5604  5618  5622  5625  5638  5648  5658  5668
        5671  5680  5684  5696  5704  5727  5730  5744  5775  5784  5796  5813  5822  5842  5891
        5895  5925  5931  5934  5941  5956  5957  5961  5962  5963  5964  5966  5969  5970  5974
        5975  5979  5980  5986  5989  5994  5997  6002  6006  6011  6012  6022  6025  6033  6034
```

	6039	6040	6045	6046	6056	6060	6068	6071	6076	6081	6082	6091	6094	6099	6102
	6113	6121	6125	6155	6170	6171	6174	6178	6179	6183	6184	6197	6198	6202	6203
	6206	6209	6220	6309	6316	6326	6327	6329	6333	6335	6337	6348	6349	6357	6364
	6371	6383	6388	6389	6396	6401	6469	6504	6508	6512	6518	6522	6540	6543	6553
	6560	6561	6562	6611	6626	6629	6638	6721	6729	6739	6865	6892	6909	6979	7094
	7109	7114	7125	7270	7277	7325	7359	7373	7468	7476	7478	7727	7730	7750	7759
	7765	7783	7792	7796	7806	7810	7813	7819	7831	7852	7864	7873	7880	7884	7898
	7890	7894	7909	7912	7917	7925	7931	7934	7936	7938	7942	7961	7971	7975	7979
	7987	7996	8000	8011	8018	8024	8028	8042	8048	8052	8062	8073	8076	8080	8088
	8095	8097	8100	8108	8117	8121	8131	8133	8134	8149	8163	8177	6183	8187	8189
	8190	8205	8212	8213	8214	8217	8222	8223	8236	8267	8274	9305	8306	8307	8316
	8329	8331	8341	8345	8347	8356	8363	8377	8387	8391	8403	8407	8409	8416	8435
	8441	8443	8444	8447	8458	8465	8471	8485	8488	8500	8502	8508	8515	8526	8537
	8548	8551	8556	8563	8567	8568	8628	8641	8647	8663	8684	8767	8783		
MOV	3893	3895	3896	3897	3898	3901	3905	3907	3908	3909	3913	3914	3915	3916	3917
	3920	3921	3922	3923	3931	3932	3933	3944	3947	3953	3964	3965	3972	3973	3979
	3984	3996	3999	4000	4010	4014	4017	4021	4025	4029	4033	4037	4041	4045	4057
	4062	4074	4075	4078	4081	4089	4091	4100	4102	4104	4132	4133	4140	4147	4154
	4164	4184	4187	4190	4192	4196	4197	4199	4213	4220	4231	4235	4238	4241	4244
	4245	4254	4255	4261	4264	4269	4272	4275	4278	4279	4288	4289	4294	4301	4302
	4305	4310	4312	4324	4330	4333	4337	4338	4524	4529	4537	4540	4541	4545	4607
	4622	4663	4664	4710	4711	4752	4753	4788	4789	4816	4817	4886	4916	4917	4956
	4989	5013	5014	5016	5018	5019	5026	5028	5056	5057	5079	5080	5125	5126	5127
	5132	5156	5160	5163	5173	5185	5186	5187	5189	5192	5200	5205	5206	5209	5221
	5223	5230	5239	5247	5249	5255	5261	5262	5271	5279	5300	5301	5304	5317	5318
	5323	5369	5376	5414	5415	5417	5422	5426	5431	5457	5461	5463	5470	5472	5483
	5485	5486	5494	5495	5496	5498	5506	5507	5511	5512	5517	5519	5521	5522	5523
	5524	5534	5537	5539	5540	5559	5560	5563	5565	5573	5574	5575	5580	5581	5536
	5587	5590	5591	5592	5595	5602	5603	5607	5621	5627	5634	5644	5654	5660	5666
	5673	5674	5676	5681	5692	5695	5698	5703	5726	5737	5741	5742	5745	5752	5753
	5754	5755	5757	5762	5767	5785	5786	5787	5795	5801	5802	5806	5810	5834	5835
	5837	5861	5863	5912	5918	5919	5920	5929	5930	5933	5935	5937	5940	5955	5968
	5993	5996	6001	6010	6013	6021	6024	6032	6055	6059	6067	6070	6073	6080	6083
	6090	6093	6095	6098	6100	6113	6120	6124	6139	6168	6173	6176	6181	6195	6200
	6205	6208	6243	6248	6255	6256	6262	6267	6268	6269	6274	6275	6276	6277	6278
	6281	6284	6285	6286	6287	6293	6298	6300	6301	6314	6321	6347	6357	6364	6371
	6383	6384	6395	6399	6402	6437	5440	6467	6468	6475	6476	6492	6493	6525	6532
	6536	6537	6546	6558	6559	6563	6574	6583	6585	6587	6596	6602	6610	6613	6625
	6628	6631	6648	6650	6674	6680	6685	6688	6694	6695	6696	6697	6698	6703	6708
	6709	6711	6715	6717	6720	6727	6751	6752	6755	6776	6786	6813	6849	6866	6867
	6868	6870	6875	6876	6884	6889	6890	6891	6893	6902	6903	6905	6906	6910	6911
	6912	6913	6923	6924	6925	6974	6976	6978	6980	6981	6982	6995	7011	7013	7015
	7018	7019	7029	7033	7034	7045	7046	7047	7053	7060	7078	7079	7080	7081	7091
	7092	7093	7104	7107	7108	7112	7113	7121	7122	7131	7133	7134	7157	7165	7173
	7175	7176	7189	7198	7206	7222	7223	7229	7232	7233	7262	7263	7267	7273	7316
	7317	7322	7345	7346	7347	7349	7375	7394	7395	7409	7411	7422	7452	7453	7454
	7455	7468	7469	7470	7471	7472	7473	7474	7475	7476	7477	7478	7479	7728	7729
	7731	7732	7736	7737	7740	7741	7742	7743	7744	7745	7746	7747	7748	7755	7764
	7785	7788	7799	7800	7905	7809	7816	7833	7834	7849	7850	7853	7856	7863	7872
	7886	7910	7920	7937	7945	7947	7962	7963	7964	7965	7968	7969	7970	7974	7978
	7982	7984	7985	7986	7994	7999	8005	8006	8010	8017	8041	8058	8065	8072	8083
	8087	8094	8095	8106	8109	8114	8122	8129	8132	8147	8148	8162	8184	8185	8197
	8199	8201	8204	8219	8224	8225	8240	8241	8250	8258	8260	8269	8282	8314	8319
	8320	8321	8327	8330	8334	8344	8346	8359	8372	8378	839C	8393	8411	8412	8413

```
        8414  8415  8422  8424  8429  8473  8483  8484  8486  8487  8497  8503  8512  8519  8524
        8547  8549  8555  8561  8580  8581  8582  8584  8587  8588  8602  8609  8610  8622  8623
        8624  8635  8638  8646  8652  8666  8668  8671  8678  8680  8693  8694  8695  8696  8705
        8768  8773  8774  8775  8778  8779  8780  8794  8811  8815  8832  8845  8849  8873  8874
        8875  8876  9877  8878  8891  8893  8894  8895  8896  8897
MOVS    4001  4186  4188  4189  4191  4193  4194  4328  4329  4334  4336  4535  4536  4560  4562
        4583  4961  5021  5072  5147  5334  5335  5345  5346  5356  5357  5366  5367  5368  5370
        5571  5630  5640  5650  5697  5714  5721  5738  5748  5749  5750  5751  5783  5793  5794
        5807  5808  5811  5812  5814  5823  5824  5890  5894  5914  5973  5978  6005  6038  6044
        6154  6240  6245  6250  6398  6445  6449  6454  6458  6464  6474  6535  6552  6566  6597
        6621  6736  6746  6781  6810  6881  7022  7028  7042  7043  7044  7048  7051  7052  7071
        7141  7182  7264  7281  7319  7329  7355  7361  7369  7398  7415  7420  7426  7431  7442
        7450  7468  7470  7471  7475  7477  7479  7784  7789  7818  7827  7836  7851  7855  7877
        7944  7990  7995  8007  8027  8038  8040  8051  8060  8061  8085  8158  8182  8239  8266
        8270  8281  8318  8362  8364  8369  8370  8392  8408  8425  8457  8528  8604  8633  8667
        8699  8704
NEG     3906  4198  4542  5756  5803  6441  6926  6927  7049  7470  7471
NOP     4249  4296  4432  4433  5460  5516  6895  8436
ROL     5245  6934  6935  6936  6937  6962  7055  7057  7058  7059  7061  7148  7150  7152  7470
        7473  7822  8208  8335  8350  8360
ROLB    8277
ROR     3936  6956  6996  6997  6998  6999  7477
RTI     6477  7117  7269  7314  7324  7378  7400  7456  7468  7470  7471  7472  8192
RTS     4345  4362  4386  4421  4434  4461  4502  4517  4573  4591  4615  4643  4647  4652  4669
        4673  4683  4698  4716  4720  4740  4758  4762  4775  4781  4794  4798  4803  4822  4834
        4843  4858  4874  4894  4905  4922  4934  4962  4983  5001  5007  5010  5051  5101  5134
        5148  5155  5166  5179  5201  5283  5330  5339  5352  5361  5378  5404  5408  5427  5451
        5471  5518  5558  5583  5614  5699  5722  5733  5763  5771  5816  5825  5864  5883  5907
        5936  5945  5982  5988  5998  6008  6014  6027  6048  6062  6078  6084  6104  6115  6127
        6133  6143  6158  6190  6215  6221  6251  6263  6294  6322  6339  6391  6403  6419  6420
        6421  6422  6423  6433  6526  6542  6545  6573  6575  6589  6612  6670  6699  6756  6790
        6805  6822  6830  6885  6896  6914  6965  6968  6983  7007  7035  7082  7096  7126  7167
        7208  7234  7282  7330  7350  7469  7473  7474  7475  7476  7477  7478  7479  7766  7832
        7887  7939  8045  8081  8086  8107  8135  8176  8178  8245  8268  8279  8290  8418  8474
        8550  8592  8611  8636  8653  8655  8685  8706  8784  8796  8818  8834  8851  8879  8898
SBC     6928  7475
SEV     6967
SUB     3934  3939  3941  4293  5029  5044  5157  5188  5214  5219  5309  5311  5314  5424  5432
        5473  5568  5635  5645  5655  6074  6089  6387  6714  7194  7458  7471  7474  7475  7991
        8166  8463
SWAB    8159  8601  8631  8698
TRAP    7479  7480
TST     3942  3974  3986  4047  4055  4094  4128  4134  4141  4148  4168  4180  4209  4211  4214
        4223  4236  4242  4250  4270  4276  4291  4303  4308  4322  4339  4349  4506  4547  4564
        4585  4624  4970  4972  4974  4987  5038  5041  5060  5070  5086  5118  5149  5170  5211
        5241  5253  5272  5306  5350  5374  5377  5386  5394  5396  5407  5439  5532  5669  5705
        5718  5728  5776  5778  5799  5847  5852  5854  5859  5878  5908  5938  6258  6271  6282
        6331  6372  6419  6420  6421  6422  6423  6427  6550  6564  6577  6723  6732  6737  6742
        6773  6834  6894  6938  6951  7006  7066  7105  7156  7159  7200  7266  7303  7307  7309
        7318  7321  7348  7357  7377  7418  7429  7451  7468  7469  7470  7471  7474  7479  7830
        7883  7913  7927  7929  8078  8092  8118  8174  8175  8210  8215  8230  8289  8299  8352
        8384  8405  8419  8439  8455  8461  8468  8520  6542  8557  8564  8594  8598  8651  8654
        8664  8681  8702  8817  8892
TSTB    3937  3988  4086  4216  4239  4273  4938  5076  5082  5120  5135  5780  5804  5845  6302
```

```
                6310  6470  6472  6538  6548  6554  6598  6622  6632  6639  6643  6651  6657  6765  7025
                7135  7177  7224  7258  7279  7327  7396  7469  7471  7757  7760  7857  7859  7861  7866
                7868  7870  7878  7896  7915  8112  8251  8253  8263  8291  8294  8301  8303  8309  8324
                8337  8367  8379  8385  8396  8432  8437  8491  8493  8495  8510  8531  8533  8540  8543
                8545  8674  8829
.ASCII          3832  9451  9454  9531  9538  9540  9544  9545  9556  9566  9567
.ASCIZ          3832  7459  7460  7469  9025  9026  9027  9028  9029  9030  9048  9049  9050  9051  9052
                9053  9054  9378  9380  9382  9384  9386  9388  9390  9392  9394  9396  9398  9400  9402
                9404  9406  9408  9410  9412  9414  9416  9418  9420  9422  9424  9426  9428  9430  9432
                9434  9436  9438  9439  9441  9443  9445  9447  9449  9452  9455  9457  9489  9490  9491
                9492  9493  9494  9495  9496  9497  9498  9499  9500  9501  9502  9503  9504  9505  9506
                9507  9508  9509  9510  9511  9512  9513  9514  9515  9516  9517  9518  9519  9520  9521
                9522  9523  9524  9525  9526  9527  9528  9529  9530  9532  9533  9534  9535  9536  9537
                9539  9541  9542  9543  9546  9547  9548  9549  9550  9551  9552  9553  9554  9555  9557
                9558  9559  9560  9561  9562  9563  9564  9565  9568  9569  9570  9571  9572  9573  9574
                9575  9576  9577  9578  9579  9580  9581  9582  9583  9584  9585  9631  9632  9633  9634
                9635  9637  9638  9639  9640  9641  9642  9643  9644  9645  9646  9647  9648  9649  9650
                9651  9652  9653  9654  9655  9656
.BLKB            7383  7461  7475  7477
.BLKW           7471  8752  8753  8754  8755  8756  8757  8758  8759  8986  9666
.BYTE           3832  5103  5104  5113  5114  6479  6480  6481  6482  6483  6484  6485  7083  7084  7085
                7086  7379  7380  7457  7458  7470  7511  7512  7513  7514  7515  7516  7517  7518  7525
                7526  7527  7528  7529  7530  7531  7532  7551  7552  7553  7554  7555  7556  7557  7558
                7564  7565  7566  7567  7568  7569  7570  7571  7591  7597  7604  7605  7606  7607  7608
                7609  7610  7611  7616  7617  7618  7619  7620  7621  7622  7623  7663  7664  7665  7666
                7667  7668  7669  7670  8709  8710  8711  8712  8713  8714  8715  8716  8980  8996  9018
                9019  9020  9021  9022  9023  9032  9033  9034  9035  9036  9037  9038  9481  9483  9485
.ENABL             4  3666
.END            9674
.ENDC           3678  3679  3693  3695  3699  3722  3832  3883  3884  3891  3893  4119  4123  5866  5870
                6223  6227  7211  7236  7284  7332  7387  7402  7464  7468  7469  7470  7471  7472  7473
                7474  7475  7476  7477  7478  7479  7480  7483  7493  8900  8904  8988  8992  9087  9091
                9372  9376  9660  9664  9668
.EQUIV          3693
.EVEN           3832  6486  7385  7462  7469  9056  9459  9487  9586  9658
.IF             3678  3679  3693  3695  3699  3722  3832  3883  3884  3891  3893  4119  4123  5866  5870
                6223  6227  7211  7236  7284  7332  7387  7402  7464  7468  7469  7470  7471  7472  7473
                7474  7475  7476  7477  7478  7479  7480  7483  7493  8900  8904  8988  8992  9087  9091
                9372  9376  9660  9664  9668
.IFF            3679  3693  3695  3699  3832  3884  3891  3893  4119  4123  5866  5870  6223  6227  7211
                7236  7284  7332  7387  7402  7464  7468  7469  7470  7471  7472  7473  7474  7475  7476
                7477  7478  7479  7483  7493  8900  8904  8988  8992  9087  9091  9372  9376  9660  9664
                9668
.IFT            7468  7474
.IFTF           7468  7474
.IIF            3678  3679  3680  3681  3682  3683  3684  3685  3686  3687  3688  3689  3690  3691  3692
                3722  3832  3893  7468  7469  7479  7480
.IRP            3832  3883  4049  4456  4603  4617  4949  5125  5132  5267  5275  5414  5426  5834  5863
                6109  6298  6321  6351  6358  6365  6378  6385  6395  6402  6492  6525  6703  6755  6902
                6912  6974  6982  7468  7471  7472  7473  8968  9013  9467  9472  9477
.LIST              2  3664  3676  3693  3722  3832  3883  7479  7480
.MACRO            39    81   164   300   478   535   684   738   825   856   904   916   956   985  1018
                1031  1052  1065  1094  1140  1177  1224  1257  1287  1343  1351  1399  1575  1773  1928
                2016  2094  2191  2269  2354  2568  2660  2736  2837  2964  3021  3083  3201  3239  3303
```

DZPPNB.P11 CROSS REFERENCE TABLE

```
        3401    3486    3525    3588    3626    3679    3725    3832    6404    7479
.MCALL  3670    3671    3672    3673    3674    3693
.NLIST     1       3    3665    3693    3722    3832    3883    7479    7480
.PAGE   3832
.REPT   3722    8980    9008    9113
.SBTTL  3679    3693    3697    3722    3832    3886    4121    5868    6225    7213    7238    7286    7334    7466    7468
        7469    7470    7471    7472    7473    7474    7475    7476    7477    7478    7479    7485    8902    8990    9089
        9374    9662
.TITLE  3678
.WORD   3722    3832    3971    4065    4107    5107    5108    5109    5110    5111    5112    5285    5286    5287    5288
        5289    5290    5291    5292    5293    5294    5453    5906    5947    5948    6141    6488    6662    6758    6792
        6793    6803    7032    7037    7095    7110    7115    7162    7203    7231    7381    7382    7469    7470    7473
        7474    7475    7476    7478    7501    7502    7503    7504    7538    7539    7540    7541    7542    7543    7544
        7545    7577    7585    7631    7639    7644    7645    7646    7647    7648    7649    7650    7651    7657    7675
        7679    7680    7683    7685    7687    7689    8585    8586    8605    8607    8639    8640    8648    8720    8721
        8722    8723    8724    8725    8726    8727    8731    8732    8733    8734    8735    8736    8737    8738    8740
        8741    8742    8743    8744    8745    8746    8747    8748    8980    8984    8985    8994    8998    9000    9002
        9004    9005    9007    9010    9015    9040    9041    9042    9043    9044    9045    9046    9060    9061    9062
        9063    9064    9065    9066    9067    9069    9070    9071    9072    9073    9074    9075    9076    9078    9079
        9080    9081    9082    9083    9084    9085    9093    9094    9095    9096    9097    9098    9099    9100    9101
        9102    9103    9104    9105    9106    9107    9108    9109    9110    9112    9115    9117    9118    9119    9120
        9121    9122    9123    9124    9125    9126    9127    9128    9129    9130    9131    9132    9134    9135    9136
        9137    9138    9139    9140    9141    9142    9143    9144    9145    9146    9147    9148    9149    9151    9152
        9153    9154    9155    9156    9157    9158    9159    9160    9161    9162    9163    9164    9165    9166    9168
        9169    9170    9171    9172    9173    9174    9175    9176    9177    9178    9179    9180    9181    9182    9183
        9185    9186    9187    9188    9189    9190    9191    9192    9193    9194    9195    9196    9197    9198    9199
        9200    9202    9203    9204    9205    9206    9207    9208    9209    9210    9211    9212    9213    9214    9215
        9216    9217    9219    9220    9221    9222    9223    9224    9225    9226    9227    9228    9229    9230    9231
        9232    9233    9234    9236    9237    9238    9239    9240    9241    9242    9243    9244    9245    9246    9247
        9248    9249    9250    9251    9253    9254    9255    9256    9257    9258    9259    9260    9261    9262    9263
        9264    9265    9266    9267    9268    9270    9271    9272    9273    9274    9275    9276    9277    9278    9279
        9280    9281    9282    9283    9284    9285    9287    9288    9289    9290    9291    9292    9293    9294    9295
        9296    9297    9298    9299    9300    9301    9302    9304    9305    9306    9307    9308    9309    9310    9311
        9312    9313    9314    9315    9316    9317    9318    9319    9321    9322    9323    9324    9325    9326    9327
        9328    9329    9330    9331    9332    9333    9334    9335    9336    9338    9339    9340    9341    9342    9343
        9344    9345    9346    9347    9348    9349    9350    9351    9352    9353    9355    9356    9357    9358    9359
        9360    9361    9362    9363    9364    9365    9366    9367    9368    9369    9370    9461    9463    9465    9470
        9475    9479    9588    9589    9590    9591    9592    9593    9594    9595    9596    9597    9598    9599    9600
        9601    9602    9603    9604    9605    9606    9607    9608    9609    9610    9611    9612    9613    9614    9615
        9616    9617    9618    9619    9620    9621    9622    9623    9624    9625    9626    9627    9628    9629    9670
```

ERRORS DETECTED: 0

```
*DZRPNB,DZRPNB/CRF=SYSMAC.SMAA,DZRPNB
RUN-TIME:  66 77 13 SECONDS
CORE USED:  25K
```

JO1

Spooler runtime 7 Seconds, 87 KCS, 936 disk reads, 0 disk writes, 240 pages printed