RP11E,RP02

MULTI-DRIVE EXERCISER

CZRP1C0

AH-9257C-MC

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA

```
.REM %
```

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

PRODUCT CODE: AC-9256C-MC

PRODUCT NAME: CZRP1C0 RP11E MUDR EXER

PRODUCT DATE: OCTOBER 25, 1978

MAINTAINER: DIAGNOSTIC ENGINEERING

CONTENTS

1. ABSTRACT

THE RP11E MULTI-DRIVE EXERCISER FROGRAM EXERCISES 1 TO 8 RP02, RPR02, OR RP03 DISK DRIVES ATTACHED TO THE SAME RP11E. IF 2 OR MORE DISK DRIVES ARE BEING EXERCISED, OPERATIONS ON THE DRIVES ARE OVERLAPPED (OTHER DRIVES ARE PERFORMING SEEK OPERATIONS WHILE ONE DRIVE IS FERFORMING A DATA TRANSFER OR WRITE CHECK OPERATION). OPERATIONS AMONG THE DRIVES ARE OPTIMIZED SO THAT A HIGH SUBSYSTEM DATA TRANSFER RATE OR A HIGH DRIVE POSITIONING RATE CAN BE MAINTAINED.

THE PERFORMANCE OF EACH DISK DRIVE IS MONITORED BY THE PROGRAM. IF A DRIVE EXCEEDS A PRESET NUMBER OF ERRORS, THE DRIVE IS AUTOMATICALLY DEASSINGED. (THE OPERATOR HAS THE OPTION OF OVER-RIDING THE AUTOMATIC DEASSIGNMENT FEATURE.) THE PROGRAM REPORTS PERFORMANCE DATA FOR EACH DRIVE BEING EXERCISED ON REQUEST FROM THE OPERATOR OR AUTOMATICALLY AT AN INTERVAL DETERMINED BY THE OPERATOR AT PROGRAM STARTUP.

ALL DATA TRANSFER COMMANDS ARE USED (I.E., READ & WRITES WITH AND WITHOUT IMPLIED SEEKS) AS WELL AS WRITE CHECK COMMANDS.

THE DATA TRANSFER COMMANDS ARE SELECTED RANDOMLY EXCEPT FOR THE WRITE CHECK COMMAND. THE WRITE CHECK COMMAND IS USED TO VERIFY A PREVIOUS WRITE OPERATION.

PROGRAM/OPERATOR COMMUNICATIONS ARE THROUGH THE TELETYPE; PROGRAM OPTIONS ARE SELECTED BY SWITCH REGISTER SETTINGS.

ALL COMMANDS, DATA PATTEPNS, 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 (WITH OR WITHOUT HARDWARE SWITCH REGISTER)
8K MEMORY
TELETYPE

```
                                        PROGRAM LOADING DEVICE
                                        KW11-L OR KW11-P CLOCK
                                        RP11E WITH 1 RP02-RPR02 OR 1 RP03

                                            OPTIONAL
                                            --------

                                        4K TO 20K ADDITIONAL MEMORY
                                        1 TO 7 ADDITIONAL RP02'S-RPR02'S OR RP03'S ON THE SAME RP11E


                2.2     MEDIA

                        THE RP11E MULTI-DRIVE 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 RP11E FORMATTER
                        PROGRAM (MAINDEC-11-DZRP2) OR BY THE 'W' COMMAND OF THE RP11E
                        MULTI-DRIVE EXERCISER (SEE SECTION 4.4).

                2.3     PRELIMINARY PROGRAMS

                        MAINDEC-11-DZRPW - RP11E DISKLESS CONTROLLER TEST
                        CZRPYC RP11E FCTNL LGC & R/W
                        MAINDEC-11-DZRPZ - RP11E DRIVE POSITIONING TEST
                        MAINDEC-11-DZRP2 - RP11E DRIVE FORMATTER PROGRAM


        3.      OPERATING THE PROGRAM
                ---------------------

                *** BEFOR STARTING REFER TO SECTION 4.2 ***
        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.  IF THE PROGRAM IS BEING RUN ON A PROCESSOR WITH 8K OF MEMORY,
                THE PROGRAM WILL NOT PRESERVE THE 'XXDP' LOADER; THE 'ABS' LOADER
                (OR THE 'ABS' LOADER SECTION OF THE 'XXDP' LOADER)  WILL BE PRESERVED,
                HOWEVER.

                THE 'XXDP' LOADER IS NOT PRESERVED IN AN 8K PROCESSOR BECAUSE OF THE
                MEMORY REQUIREMENTS OF THE 'XXDP' LOADER;  THE PROGRAM USES THE
                'XXDP' LOADER SPACE FOR BUFFER SPACE ON THE 8K PROCESSOR.

                THE PROGRAM MAY BE INCLUDED IN AN 'XXDP' CHAIN IF THE PROCESSOR TO
                RUN THE CHAIN IS 12K OR GREATER.

        3.2     THE NORMAL STARTING LOCATION FOR THE PROGRAM IS 200(8).  PARAMETERS
                MAY BE CHANGED BY STARTING THE PROGRAM AT LOCATION 204(8).

        3.3     THE PROGRAM MAY BE RESTARTED FROM EITHER LOCATION 200(8) OR
                OR LOCATION 204 (8).

        3.4     SET THE 'FORMAT' AND 'MAINTENANCE' SWITCHES ON THE RP11E TO 'NORMAL';
                SET THE CONTROLLER WRITE LOCKOUT AND DRIVE WRITE ENABLE SWITCHES AS
                REQUIRED.
```

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

IF THE PROGRAM IS BEING RUN AS PART OF AN 'XXDP' CHAIN, THE 'T' COMMAND IS NOT REQUIRED TO START THE TEST: THE PROGRAM WILL AUTOMATICALLY INITIATE A 'TA' COMMAND. IF THE 'XXDP' CHAIN IS BEING LOADED FROM AN RP02, RPR02, OR RP03, DRIVE 0 WILL BE BYPASSED WHEN THE 'TA' COMMAND IS ISSUED. NOTE THAT OPERATOR INTERVENTION IS NOT REQUIRED.

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 'SEKMOD'.

A. IF PARAMETER 'SEKMOD' IS 0, END OF PASS OCCURS WHEN THE DRIVE HAS READ 6.25 X 10^8 WORDS (1 X 10^10 BITS).
B. IF PARAMETER 'SEKMOD' IS 1, END OF PASS OCCURS WHEN THE DRIVE HAS PERFORMED 1 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 ERROP OCCURS.

3.6 RUN TIME

THE EXERCISER PROGRAM MAY BE RUN IN TWO MODES. THE MODE IS DETERMINED BY THE VALUE IN PARAMETER 'BUFRSZ'. IF 'BUFRSZ' IS '1' SECTOR, THE PROGRAM RUNS IN A SEEK HEAVY MODE; IF 'BUFRSZ' APPROACHES ONE TRACK IN SIZE (5000 OCTAL) 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 8K, THE READ/WRITE RATIO PARAMETER - 'RATIO', AND BY SWITCHES 0 & 1.

3.6.1 RUN TIME IN 'DATA TRANSFER MODE'

14 HOURS FOR 1 DRIVE.

(ADD 4 HOURS FOR EACH ADDITIONAL DRIVE TESTED.)

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

12 HOURS FOR 1 DRIVE

(ADD 3 HOURS FOR EACH ADDITIONAL DRIVE TESTED.)

3.6.2 RUN TIME IN 'SEEK VERIFICATION MODE'

ASSUME THE FOLLOWING PARAMETERS VALUES:

PARAMETER 'BUFRSZ' = 1 SECTOR (400(8) WORDS)
PARAMETER 'LT' = 'FT'
PARAMETER 'LS' = 'FS'
SW<00> = 1 (READ ONLY MODE)

17 HOURS FOR 1 DRIVE

(ADD 8 HOURS FOR EACH ADDITIONAL DRIVE TESTED.)

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
RP11E	176710	254
TTY PRINTER	177564	NOT USED
TTY KEYBOARD	177560	60
KW11-L	177546	100
KW11-P	172542	104

3.8 SUBSYSTEMS WITH BOTH RP02-RPR02 AND RP03 DRIVES

RP11E SUBSYSTEMS WITH RP02-RPR02 AND RP03 DISK DRIVES ARE NOT SUPPORTED BY THE MULTI-DRIVE EXERCISER. IF THE PROGRAM IS RUN ON A SYSTEM WITH BOTH DRIVE TYPES, EITHER THE RP02-RPR02 DRIVES OR THE RP03 DRIVES MUST BE POWERED OFF.

4. CONTROLLING THE PROGRAM

4.1 PARAMETERS

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, PRELOADED PARAMETERS WILL BE ASSUMED. IF THE PROGRAM IS STARTED FROM LOCATION 204, THE FOLLOWING MESSAGE WILL BE TYPED.

ENTER PARAMETERS:

THE PROGRAM WILL THEN IDENTIFY THE PARAMETER BY THE NAME GIVEN BELOW, DISPLAY THE CURRENT VALUE OF THE PARAMETER (IN OCTAL) AND THE
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 RETURN TO THE BEGINNING OF THE PARAMETER LIST. IF A PERIOD IS ENTERED (FOLLOWED BY A CARRIAGE RETURN) EITHER TO TERMINATE AN ENTRY OR IN PLACE OF A PARAMETER CHANGE ENTRY, 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' OR A 'PERIOD' - 'CARRIAGE RETURN'.)

4.1.1 KEYBOARD ENTRY PARAMETERS

NAME	DEFAULT VALUE	VALUE RANGE	FUNCTION
BUFRSZ	(SEE NOTE)		CONTROLS THE MAXIMUM BUFFER SIZE USED FOR DATA TRANSFERS
PASCNT	1	1 - 1747	NUMBER OF PASSES TO END OF TEST.
INTRVL	5	0 - 377	DETERMINES THE INTERVAL (IN MINUTES) BETWEEN AUTOMATIC PERFORMANCE SUMMARY TYPEOUTS
CMPLMT	4	0 - 'BUFRSZ'	THE NUMBER OF DATA COMPARSION ERRORS PRINTED OUT IF SW<02>=0
LC	(SEE NOTE)		THE MAXIMUM CYLINDER ADDRESS
FC	(SEE NOTE)		THE MINIMUM CYLINDER ADDRESS. VALUE MUST NOT BE GREATER THAN VALUE IN 'LC'
LT	23	0 - 23	THE MAXIMUM TRACK ADDRESS
FT	0	0 - 23	THE MINIMUM TRACK ADDRESS THE MINIMUM TRACK ADDRESS MUST NOT BE GREATER THAN THE VALUE IN 'LT'
LS	11	0 - 11	THE MAXIMUM SECTOR ADDRESS
FS	0	0 - 11	THE MINIMUM SECTOR ADDRESS THE MINIMUM SECTOR ADDRESS MUST NOT BE GREATER THAN THE VALUE IN 'LS'
SEKMOD	1	0 OR 1	IF EQ 0, END OF PASS DETERMINED BY THE 'WORDS READ' COUNT. IF EQ 1, END OF PASS IS DETERMINED BY THE NUMBER OF SEEKS.
RPADR	176710	N/A	THE RP11E UNIBUS ADDRESS
RPVEC	254	N/A	THE RP11E VECTOR ADDRESS
RATIO	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

                    AJTOCK      1     0 OR 1        IF EQ 1, THE PROGRAM
                                                    PERFORM WRITE CHECKS AFTER
                                                    EACH WRITE COMMAND.
                                                    IF EQ 0, THE PROGRAM
                                                    WILL PERFORM WRITE CHECKS
                                                    RANDOMLY.

                    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 5400 (8) 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 'BUFRSZ' DETERMINED BY THE PROGRAM.

                           THE MAXIMUM VALUES FOR 'FC' AND 'LC' ARE THE DETERMINED BY
                           THE DRIVE TYPE BEING TESTED: 202 (10) FOR RP02'S; 405 (10) FOR
                           RP03'S; AND 405 (10) FOR SYSTEMS WITH BOTH RP02'S AND RP03'S.


4.1.2   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
        ---     ---     --------        --------

        1216    $LKCSR  172540          ADDRESS OF KW11-P STATUS REGISTER
        1220    $LKCSB  172542          ADDRESS OF KW11-P COUNTER BUFFER
        1222    $LPVEC     104          KW11-P VECTOR ADDRESS
        1224    $LKS    177546          ADDRESS OF KW11-L STATUS REGISTER
        1226    $LLVEC     100          KW11-L VECTOR ADDRESS
        1232    HZ          74          74 (60 DECIMAL) IF SYSTEM IS 60 HZ;
                                        62 (50 DECIMAL) IF SYSTEM IS 50 HZ.

4.2     SWITCH REGISTER SETTINGS

                IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH
                REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS
                THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER.
                IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES
                AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH
                REGISTER (LOC. 176) IS USED.
```

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

1) TYPE CONTROL G <^G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW=" HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
   IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

```
SW <15> - 1     HALT ON ERROR
SW <13> = 1     INHIBIT ERROR TYPEOUT
SW <10> - 1     RING THE TELETYPE BELL IF ERROR
SW <6>  = 1     DON'T SYSTEM STATUS WHEN PROGRAM STARTED
                INHIBIT PERFORMANCE SUMMARY TYPEOUTS
SW <5>  = 1     LOOP ON PRESENT OPERATION(S)
SW <4>  = 1     INHIBIT MAXIMUM ERROR COUNT CHECK; DO NOT DEASSIGN
                DRIVES WHEN NORMAL END OF TEST REACHED.
SW <3>  - '     DISPLAY SECTOR WHICH GAVE 'HARD' ERROR
                IF DATA COMPARE ERRORS & SW<2> SET, DISPLAY REST
                OF BUFFER
SW <2>  = 1     DISPLAY ALL DATA COMPARSION ERRORS
SW <1>    1     INHIBIT DATA COMPARSION AFTER READ ORDERS
SW <0>  = 1     READ ONLY MODE
```

4.3 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.

IF THE PROGRAM IS BEING RUN AS PART OF AN 'XXDP' PROGRAM CHAIN, OPERATIONS ARE AUTOMATICALLY STARTED. THE PROGRAM ITSELF SIMULATES THE ISSUING OF A 'TA' COMMAND.

4.3.1 '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.

4.3.2 '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 OPERATION COMPLETE.

3. IF THE 'DA' COMMAND IS USED, ONLY DRIVES BEING TESTED WILL WILL BE DEASSIGNED - THE ERROR MESSAGE IN (1) ABOVE WILL NOT BE DISPLAYED.

4.3.3 '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.

4.3.4 'W' COMMAND

USED TO WRITE A DATA PACK WITH DATA ACCEPTABLE TO THE RP02 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.
WO<CR> - GENERATE A DATA PACK ON DRIVE 0.

NOTES: 1. THE 'W' COMMAND SHOULD NOT BE USED UNLESS 'BUFRSZ' 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.

2. THE 'W' COMMAND PERFORMS A SEQUENTIAL WRITE OF THE PACK USING A 'WRITE WITH IMPLIED SEEKS' COMMAND. THE DATA PATTERN USED FOR EACH WRITE IS SELECTED RANDOMLY. THE PACK IS WRITTEN SEQUENTIALLY, BEGINNING AT 'FC', 'FT' AND CONTINUING TO 'LC', 'LT'.

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

4. 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.

5. THE DATA WRITTEN IS VERIFIED BY A 'WRITE CHECK COMMAND' (IF THE PARAMETER 'AUTOCK' IS '1').

4.3.5 'R' COMMAND

USED TO PERFORM A SEQUENTIAL READ OF THE PACK.

FORMAT: RN<CR>

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

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

NOTES: 1. THE PROGRAM WILL 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 'FC', 'FT' TO THE ADDRESS SPECIFIED
        BY 'LC', 'LT'. THE READ WILL BE SFQUENTIAL.

4.3.6 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)
?DRIVE N OFFLINE	T, W, R
?DRIVE N NOT ASSIGNED	D, S
?DRIVE N ALREADY ASSIGNED	T, W, R
?DRIVE N UNSAFE	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
'SOFT'	THE NUMBER OF SOFT DATA ERRORS
'HARD'	THE NUMBER OF HARD DATA ERRORS
'SUSI'	THE NUMBER OF 'SUSI' ERRORS
'MISP'	THE NUMBER OF POSITIONING ERRORS
'OTHER'	THE TOTAL ERRORS OF OTHER TYPES

5.2 SOFT/HARD ERROR DEFINITIONS

5.2.1 HARD ERRORS

'CSME', 'WPE', AND 'LPE' ERRORS WHICH ARE NOT RECOVERABLE AFTER

10 RETRY ATTEMPTS FOLLOWED BY A HOME SEEK AND A RE-SEEK TO THE
ERROR CYLINDER AND 10 ADDITIONAL RETRY ATTEMPTS.

5.2.2 SOFT ERRORS

'CSME', 'WPE', OR 'LPE' ERRORS WHICH ARE RECOVERABLE DURING THE
10 RETRY, HOME SEEK, RE-SEEK, 10 RETRY SEQUENCE DESCRIBED ABOVE.

6. DATA CHECKING & ERROR RECOVERY

6.1 DATA COMPARISON

DATA COMPARISON OCCURS AFTER EACH READ ORDER UNDER THE THE FOLLOWING
CONDITIONS:

A. THE ORDER TERMINATED WITH NO ERROR.

B. THE OPERATION TERMINATED WITH A DATA ERROR BUT READ CORRECTLY
CORRECTLY AFTER RETRY ATTEMPTS.

6.2 VERIFICATION OF DATA WRITTEN

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

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

(THE MESSAGE TAGS ARE GIVEN FOR REFERENCE.)

MESSAGE TAG	TEXT

```
                        EM1     RP11 DIDN'T RESPOND TO ADDRESSING

                                WHEN THE PROGRAM ADDRESSED THE RP11, A BUS TIMEOUT
                                OCCURED.

                        EM2     'INVALID RP11 COMMAND'

                                THE PROGRAM ATTEMPTED TO ISSUE AN INVALID RP11 COMMAND.
                                THIS ERROR WILL INDICATE EITHER A PROGRAMMING ERROR OR
                                A CENTRAL PROCESSOR FAILURE.

                        EM3     'UNDEFINED INTERRUPT FROM THE RP11'

                                AN INTERRUPT FROM THE RP11 OCCURED AND NO ATTENTION BITS
                                WERE SET AND NO TRANSFER WAS UNDERWAY.

                        EM10    'CAN'T MATCH DATA READ WITH A PATTERN'

                                THE DATA COMPARSION ROUTINE CANNOT IDENTIFY THE DATA
                                PATTERN READ.

                        EM12    'DATA COMPARSION ERROR'

                                THE COMPARSION ROUTINE WAS ABLE TO IDENTIFY THE DATA
                                PATTERN; HOWEVER, A COMPARSION ERROR OCCURED.

                        EM15    ''ERR' NOT SET BUT 'RPER' CONTENTS NOT ZERO'

                                THE ERROR BIT IN 'RPCS' WAS NOT SET BUT ONE OR MORE
                                ERROR BITS WAS SET IN THE ERROR REGISTER.

                        EM16    'WORD COUNT NOT ZERO OR BUFFER ADDRESS WRONG

                                AFTER AN ERROR FREE DATA TRANSFER, THE WORD COUNT REGISTER
                                DID NOT GO TO ZERO OR THE BUFFER ADDRESS REGISTER CONTENTS
                                WERE INCORRECT.
                        EM17    'SOFTWARE TIMEOUT DURING POSITIONING'

                                A SEEK OR HOME SEEK COMMAND DID NOT COMPLETE WITHIN 1 SECOND.

                        EM20    'SOFTWARE TIMEOUT DURING I/O'

                                A DATA TRANSFER OR WRITE CHECK COMMAND DID NOT COMPLETE WITHIN
                                1 SECOND.

                        EM21    'DRIVE OFFLINE'

                                THE INDICATED DRIVE HAS GONE OFFLINE.

                        EM22    'DRIVE UNSAFE'

                                THE INDICATED DRIVE HAS BECO   UNSAFE.

                        EM23    'SEEK INCOMPLETE'

                                A SEEK INCOMPLETE ERROR HAS OCCURED ON THE INDICATED
```

```
                                        DRIVE.

                                EM24    'UNIBUS TRANSFER ERROR'

                                        A UNIBUS TIMING ERROR OR NON-EXISTENT MEMORY ERROR HAS
                                        OCCURED.

                                EM25    'CONTROLLER ERROR'

                                        A 'MODE' ERROR OR A 'PROGRAM' ERROR HAS OCCURED.

                                EM26    'NON-EXISTENT DISK ADDRESS'

                                        A NON-EXISTENT SECTOR, TRACK, OR CYLINDER ERROR HAS
                                        OCCURED.

                                EM27    'DATA ERROR'

                                        ANY OR ALL OF THE FOLLOWING ERRORS OCCURED:  'CHECKSUM'
                                        ERROR, 'WORD PARITY ERROR', OR 'LONGITUDINAL PARITY' ERROR.

                                EM32    'WRITE CHECK ERROR (NO DATA ERROR)'

                                        A WRITE CHECK ERROR OCCURED.  'CHECKSUM', 'WORD PARITY
                                        ERROR', OR 'LONGITUDINAL PARITY' ERROR BITS WERE NOT SET.

                                EM33    'WRITE CHECK ERROR (DATA ERROR BITS SET)'

                                        A WRITE CHECK ERROR OCCURED.  'CHECKSUM', 'WORD PARITY', OR
                                        'LONGITUDINAL PARITY' ERROR BITS WERE SET.

                                EM34    'HEADER NOT FOUND'

                                        A 'HEADER NOT FOUND' ERROR OCCURED.  THE PROGRAM VERIFIED
                                        THAT THE DISK WAS ON THE CORRECT CYLINDER.


                                EM35    'FORMAT ERROR WHILE SEARCHING FOR INITIAL SECTOR'

                                        A 'FORMAT ERROR' WAS DETECTED WHILE THE CONTROLLER WAS
                                        SEARCHING FOR THE FIRST SECTOR OF THE DATA TO BE TRANSFERED.

                                EM36    'HEADER NOT FOUND, CYLINDER IN HEADER DOES NOT COMPARE'

                                        A HEADER NOT FOUND ERROR OCCURED.  THE PROGRAM READ THE
                                        HEADER OF THE SECTOR BEING SEARCHED FOR (USING A HEADER
                                        READ) AND THE CYLINDER ADDRESS FIELD IN THE HEADER DID NOT
                                        COMPARE WITH THE CYLINDER THE DISK WAS EXPECTED TO BE AT.
                                        THE PROGRAM ISSUES A HOME SEEK TO THE DRIVE AFTER REPORTING
                                        THIS ERROR.

                                EM37    'DIFFERENT ERROR DURING RETRY'

                                        DURING ERROR RECOVERY OR RETRY A DIFFERENT ERROR TYPE OCCURED.

                                EM43    ''ERR' SET BUT NO ERROR BITS SET'
```

'ERR' IN 'RPCS' WAS SET, BUT NO ERROR BITS IN 'RPER' WERE SET.

EM44 'WRITE PROTECTION VIOLATION'

THE PROGRAM ATTEMPTED TO WRITE ON A WRITE PROTECTED DRIVE.

EM45 'FORMAT ERROR DURING I/O'

A 'FORMAT ERROR' WAS DETECTED WHILE THE CONTROLLER WAS SEARCHING FOR THE SECOND OR SUBSEQUENT SECTOR TO BE TRANSFERED.

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, 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 AND IS ONLINE. THE ASSIGNMENT ROUTINE THEN ISSUES A HOME SEEK 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 (COMMAND INITIATION DESCRIPTION)

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 RP11E 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.

                            SOFTWARE TIMEOUT DURING POSITIONING - EM17
                            SOFTWARE TIMEOUT DURING I/O - EM20
                            DRIVE OFFLINE - EM21
                            DRIVE UNSAFE - EM22

                        B.  ERRORS REPORTED FOR OPERATIONS WHICH COMPLETE NORMALLY.

                            SEEK INCOMPLETE - EM23
                            UNIBUS TRANSFER ERROR - EM24
                            CONTROLLER ERROR - EM25
                            NON-EXISTENT DISK ADDRESS - EM26
                            DATA ERROR - EM27
                            WRITE CHECK ERROR (NO DATA ERROR) - EM32
                            WRITE CHECK ERROR (DATA ERROR BITS SET) - EM33
                            HEADER NOT FOUND - EM34
                            FORMAT ERROR WHILE SEARCHING FOR INITIAL SECTOR - EM35
                            POSITIONING ERROR - EM36
                            DIFFERENT ERROR DURING RETRY - EM37
                            WRITE PROTECTION VIOLATION - EM44
                            FORMAT ERROR DURING I/O - EM45


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

                            RP11 DIDN'T RESPOND TO ADDRESSING - EM1
                            INVALID RP11 COMMAND - EM2
                            UNDEFINED INTERRUPT FROM THE RP11 - EM3
                            CAN'T MATCH DATA READ WITH A PATTERN - EM10
                            DATA COMPARE ERRORS - EM12
                            'ERR' NOT SET BUT 'RPER' CONTENTS NOT ZERO - EM15
                            WORD COUNT NOT ZERO OR BUFFER ADDRESS WRONG - EM16
                            'ERR' SET BUT NOT ERROR BITS IN 'RPER' SET - EM43

                8.2     SELECTION OF OPERATION VARIABLES

                        A.  SECTOR ADDRESS SELECTION IS RANDOM BETWEEN THE VALUES IN
                            'FS' AND 'LS'.  TRACK ADDRESS SELECTION IS RANDOM
                            BETWEEN THE VALUES IN 'FT' AND 'LT'.  CYLINDER ADDRESS
                            SELECTION IS RANDOM BETWEEN 'FC' AND 'LC'.

                        B.  THE BUFFER SIZE IS RANDOM SELECTED BETWEEN 4 (10) - AND THE
```

VALUE IN 'BUFRSZ'. 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 20 (8) STANDARD PATTERNS.

D. THE ORDERS ARE SELECTED RANDOMLY. WRITE CHECK ORDERS ARE PERFORMED ONLY IF THE PREVIOUS ORDER WAS A WRITE COMMAND.

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.

8.3 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.

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

PAT 0	PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7
000000	000001	177776	000000	000000	052525	007417	026455
000000	000003	177774	000000	010421	052525	007417	026455
000000	000007	177770	000000	021042	052525	007417	026455
000000	000017	177760	177777	031463	125252	170360	151322
000000	000037	177740	177777	042104	125252	170360	151322
000000	000077	177700	177777	052525	125252	170360	151322
000000	000177	177600	000000	063146	052525	007417	026455
000000	000377	177400	000000	073567	052525	007417	026455
000000	000777	177000	177777	104210	125252	170360	151322
000000	001777	176000	177777	114631	125252	170360	151322
000000	003777	174000	000000	125252	052525	007417	026455
000000	007777	170000	177777	135673	125252	170360	151322
000000	017777	160000	000000	146314	052525	007417	026455
000000	037777	140000	177777	156735	125252	170360	151322
000000	077777	100000	000000	167356	052525	007417	026455
000000	177777	000000	177777	177777	125252	170360	151322

PAT 8	PAT 9	PAT 10	PAT 11	PAT 12	PAT 13	PAT 14	PAT 15

```
                                ------  ------  ------  ------  ------  ------  ------
                                077577  000001  177776  172666  077777  153333  000000  177777
                                077577  000002  177775  155555  137777  066667  177777  000000
                                077577  000004  177773  172666  157777  153333  177777  000000
                                077577  000010  177767  155555  167777  066667  177777  000000
                                077577  000020  177757  172666  173777  153333  177777  000000
                                077577  000040  177737  155555  175777  066667  177777  000000
                                077577  000100  177677  172666  176777  153333  177777  000000
                                077577  000200  177577  155555  177377  066667  177777  000000
                                077577  000400  177377  172666  177577  153333  177777  000000
                                077577  001000  176777  155555  177677  066667  177777  000000
                                077577  002000  175777  172666  177737  153333  177777  000000
                                077577  004000  173777  155555  177757  066667  177777  000000
                                077577  010000  167777  172666  177767  153333  177777  000000
                                077577  020000  157777  155555  177773  066667  177777  000000
                                077577  040000  137777  172666  177775  153333  177777  000000
                                077577  100000  077777  155555  177776  066667  177777  000000


9.      PROGRAM LISTING
        ---------------
%


        .NLIST  MC,MD,CND
        .LIST   ME


.TITLE  CZRP1C0, RP11 MUDR EXER
;*COPYRIGHT (C) 1975,1978
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY C. HESS/F. ROEMER
;*
;*THIS PROGRAM WAS ASSEMBLED USING THF PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
;*
.SBTTL  OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH                  USE
;*      ------          --------------------
;*        15            HALT ON ERROR
;*        13            INHIBIT ERROR TYPEOUTS
;*        10            BELL ON ERROR
;*         7            DON'T TYPE SYSTEM STATUS.
;*         6            INHIBIT PERFORMANCE SUMMARY TYPEOUTS
;*         5            DON'T CHANGE PARAMETERS (LOOP ON PRESENT VALUES)
;*         4            DO NOT CHECK FOR MAXIMUM ERROR COUNT OR DEASSIGN
;*                      DRIVE WHEN NORMAL END OF TEST REACHED
;*         3            DISPLAY SECTOR WHICH GAVE 'HARD' ERROR
;*                      IF DATA COMPARE ERROR & SW2 SET, DISPLAY
;*                      REMAINDER OF BUFFER
;*         2            DISPLAY ALL DATA COMPARE ERRORS
;*         1            INHIBIT DATA COMPARSION AFTER READ ORDERS
```

```
                                ;*          0            READ ONLY MODE

                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

                                ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
```

```
1112            000010                  SW03=   10
1113            000004                  SW02=   4
1114            000002                  SW01=   2
1115            000001                  SW00=   1
1116                                    .EQUIV  SW09,SW9
1117                                    .EQUIV  SW08,SW8
1118                                    .EQUIV  SW07,SW7
1119                                    .EQUIV  SW06,SW6
1120                                    .EQUIV  SW05,SW5
1121                                    .EQUIV  SW04,SW4
1122                                    .EQUIV  SW03,SW3
1123                                    .EQUIV  SW02,SW2
1124                                    .EQUIV  SW01,SW1
1125                                    .EQUIV  SW00,SW0
1126
1127                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1128            100000                  BIT15=  100000
1129            040000                  BIT14=  40000
1130            020000                  BIT13=  20000
1131            010000                  BIT12=  10000
1132            004000                  BIT11=  4000
1133            002000                  BIT10=  2000
1134            001000                  BIT09=  1000
1135            000400                  BIT08=  400
1136            000200                  BIT07=  200
1137            000100                  BIT06=  100
1138            000040                  BIT05=  40
1139            000020                  BIT04=  20
1140            000010                  BIT03=  10
1141            000004                  BIT02=  4
1142            000002                  BIT01=  2
1143            000001                  BIT00=  1
1144                                    .EQUIV  BIT09,BIT9
1145                                    .EQUIV  BIT08,BIT8
1146                                    .EQUIV  BIT07,BIT7
1147                                    .EQUIV  BIT06,BIT6
1148                                    .EQUIV  BIT05,BIT5
1149                                    .EQUIV  BIT04,BIT4
1150                                    .EQUIV  BIT03,BIT3
1151                                    .EQUIV  BIT02,BIT2
1152                                    .EQUIV  BIT01,BIT1
1153                                    .EQUIV  BIT00,BIT0
1154
1155                                    ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
1156            000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
1157            000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
1158            000014                  TBITVEC=14              ;;''T'' BIT
1159            000014                  TRTVEC= 14              ;;TRACE TRAP
1160            000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
1161            000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1162            000024                  PWRVEC= 24              ;;POWER FAIL
1163            000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1164            000034                  TRAPVEC=34              ;;''TRAP'' TRAP
1165            000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
1166            000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
1167            000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
1168
1169                                    ;RP11 REGISTER INDEX VALUES
1170
1171            000000                  RPDS=00                         ;DRIVE STATUS REGISTER
1172            000002                  RPER=02                         ;ERROR REGISTER
1173            000004                  RPCS=04                         ;CONTROL REGISTER
1174            000006                  RPWC=06                         ;WORD COUNT REGISTER
1175            000010                  RPBA=10                         ;BUFFER ADDRESS REGISTER
1176            000012                  RPCA=12                         ;CYLINDER ADDRESS REGISTER
1177            000014                  RPDA=14                         ;SECTOR/TRACK ADDRESS REGISTER
1178            000016                  RPM1=16                         ;MAINTENANCE REGISTER #1
1179            000020                  RPM2=20                         ;MAINTENANCE REGISTER #2
1180            000022                  RPM3=22                         ;MAINTENANCE REGISTER #3
1181            000024                  SUCA=24                         ;SELECTED UNIT CYLINDER ADDRESS REGISTER
1182            000026                  SILO=26                         ;SILO REGISTER
1183
1184                                    ,RP11 OP CODE DEFINITIONS
1185
1186            000001                  CLEAR=1                         ;CLEAR THE CONTROLLER
1187            000003                  WRTSEK=3                        ;WRITE WITH IMPLIED SEEK AND HEAD SELECTION
1188            000005                  RDSEK=5                         ;READ WITH IMPLIED SEEK AND HEAD SELECT
1189            000007                  WRTCHK=7                        ;WRITE CHECK WITH IMPLIIED SEEK AND HEAD SELECT
1190            000011                  SEEK=11                         ;SEEK
1191            000013                  WRITE=13                        ;WRITE (NO IMPLIED SEEK OR HEAD SELECT)
1192            000015                  HOMSEK=15                       ;HOME SEEK
1193            000017                  READ=17                         ;READ (NO IMPLIED SEEK OR HEAD SELECT)
1194
1195                                    ;RP11 REGISTER EQUATES
1196
1197                                    ;RPDS (DRIVE STATUS REGISTER)
1198
1199                                    ;ATTENTION BITS ARE REFERENCED BY BIT NUMBER (BIT00 - BIT07)
1200            000400                  SUWP=400                        ;SELECTED UNIT WRITE PROTECTED
1201            001000                  SUFU=1000                       ;SELECTED UNIT FILE UNSAFE
1202            002000                  SUSU=2000                       ;SELECTED UNIT SEEK UNDERWAY
1203            004000                  SUSI=4000                       ;SELECTED UNIT SEEK INCOMPLETE
1204            010000                  HNF=10000                       ;HEADER NOT FOUND
1205            020000                  SURP03=20000                    ;SELECTED UNIT IS AN RP03
1206            040000                  SUOL=40000                      ;SELECTED UNIT IS ONLINE
1207            100000                  SURDY=100000                    ;SELECTED UNIT IS READY
1208
1209                                    ;RPER (ERROR REGISTER)
1210
1211            000001                  DSKERR=1                        ;DISK ERROR
1212            000002                  EOP=2                           ;END OF PACK
1213            000004                  NXME=4                          ;NON-EXISTENT MEMORY
1214            000010                  WCE=10                          ;WRITE CHECK ERROR
1215            000020                  TIMEE=20                        ;TIMING ERROR
1216            000040                  CSME=40                         ;CHECKSUM ERROR
1217            000100                  WPE=100                         ;WORD PARITY ERROR
1218            000200                  LPE=200                         ;LONGITUDINAL PARITY ERROR
1219            000400                  MODERR=400                      ;MODE ERROR
1220            001000                  FMTE=1000                       ;FORMAT ERROR
1221            002000                  PROG=2000                       ;PROGRAM ERROR
1222            004000                  NXS=4000                        ;NON-EXISTENT SECTOR
1223            010000                  NXT=10000                       ;NON-EXISTENT TRACK
```

```
1224            020000                  NXC=20000                       ;NON-EXSITENT CYLINDER
1225            040000                  FUV=40000                       ;FILE UNSAFE VIOLATION
1226            100000                  WPV=100000                      ;WRITE PROTECT VIOLATION
1227
1228                                    ;RPCS  (CONTROL REGISTER)
1229
1230                                    ;'GO' BIT AND FUNCTION BITS ARE LOADED TOGETHER AND ARE DEFINED ELSEWHERE.
1231                                    ;EXTENDED MEMORY ADDRESS BITS ARE REFERED TO BY BIT NUMBER
1232            000100                  IDE=100                         ;INTERRUPT ON DONE
1233            000200                  RDY=200                         ;RP11 READY
1234                                    ;DRIVE SELECT BITS ARE REFERED TO BY BIT NUMBER
1235            004000                  HDR=4000                        ;HEADER
1236            010000                  MODE=10000                      ;MODE
1237            020000                  AIE=20000                       ;ATTENTION INTERRUPT ENABLE
1238            040000                  HE=40000                        ;HARD ERROR
1239            100000                  ERR=100000                      ;ERROR
1240
1241                                    .SBTTL  TRAP CATCHER
1242
1243            000000                          .=0
1244                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1245                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1246                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1247            000174                          .=174
1248  000174    000000                DISPREG: .WORD  0                 ;;SOFTWARE DISPLAY REGISTER
1249  000176    000000                SWREG:   .WORD  0                 ;;SOFTWARE SWITCH REGISTER
1250
1251                                    .SBTTL  ACT11 HOOKS
1252
1253                                    ;;*****************************************************************
1254                                    ;HOOKS REQUIRED BY ACT11
1255            000200                          $SVPC=.                 ;SAVE PC
1256            000046                          .=46
1257  000046    003634                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1258            000052                          .=52
1259  000052    040000                          .WORD   40000           ;;2)SET LOC.52 TO 40000
1260            000200                          .=$SVPC                 ;; RESTORE PC
1261
1262                                    .SBTTL  STARTING ADDRESSES
1263
1264            000200                          .=200
1265
1266  000200    000137  002016                  JMP     START           ;START PROGRAM WITHOUT CHANGING PARAMETERS
1267  000204    000137  002024                  JMP     START1          ;START PROGRAM AND CHANGE PARAMETERS
```

L 2

COMMON TAGS SEQ 0024

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102     000          $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000          $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000          $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001          $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134     000          $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000          $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002          $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012          $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
001157     000          $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>-0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000          $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000          $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000          $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001200  000000          $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001202  000000          $REG10: .WORD   0               ;;CONTAINS (($REGAD)+20)
001204  000000          $REG11: .WORD   0               ;;CONTAINS (($REGAD)+22)
001206  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001212     077          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001213     015          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001214  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
001216  176710          RPADR:  .WORD   176710          ;FIRST ADDRESS OF RP11
001220  000254          RPVEC:  .WORD   254             ;RP11 VECTOR ADDRESS
```

```
1324  001222  000240                  RPPRIO: .WORD   5*32.           ;RP11 PRIORITY
1325  001224  172540                  $LKCSR: .WORD   172540          ;ADDR OF KW11-P STATUS REGISTER
1326  001226  172542                  $LKCSB: .WORD   172542          ;ADDR OF KW11-P COUNTER BUFFER
1327  001230  000104                  $LPVEC: .WORD   104             ;ADDR OF KW11-P VECTOR
1328  001232  177546                  $LKS:   .WORD   177546          ;ADDR OF KW11-L STATUS REGISTER
1329  001234  000100                  $LLVEC: .WORD   100             ;ADDR OF KW11-L VECTOR
1330  001236  177777                  CLKFLG: .WORD   -1              ;'0' IF A CLOCK IS AVAILABLE
1331  001240  000074                  HZ:     .WORD   74              ;74 (8) IF 60 HZ SYSTEM; 62 (8) IF 50 HZ SYSTEM
1332  001242  000000                  CNTRLC: .WORD   0               ;VECTOR ADDRESS FOR CONTROL C
1333  001244  000000                  LSTAD:  .WORD   0               ;CONTAINS LAST MEMORY ADDRESS
1334  001246  001100                  STACK1: .WORD   STACK           ;CONSTANT USED TO CHECK STACK POINTER
1335  001250  000000                  STATIN: .WORD   0               ;ERROR RATE DATA DISPLAY INDICATOR
1336  001252  000000                  PACK:   .WORD   0               ;'W' OR 'R' COMMAND INDICATOR
1337  001254  000003                  FAIRNS: .WORD   3               ;'FAIRNESS' COUNT FOR BUFFER WAIT
1338  001256  000000                  MAXCYL: .WORD   0               ;MAXIMUM CYLINDER ADDRESS ALLOWED FOR
1339                                                                  ;ANY DRIVE TESTED.  (FILLED IN DURING STARTUP.)
1340  001260  001750                  TIMOUT: .WORD   1000.           ;TIME (IN MS) ALLOWED FOR OPERATION
1341  001262  000000                  CHNGPM: .WORD   0               ;IS NON-ZERO IF PARAMETERS TO BE CHANGED
1342  001264  000000                  DRIVE:  .WORD   0               ;STORE DRIVE NUMBER HERE
1343  001266  000000                  CYL.ER: .WORD   0               ;STORE CYLINDER ADDRESS HERE
1344  001270  000000                  TRK.ER: .WORD   0               ;STORE TRACK ADDRESS HERE
1345  001272  000000                  SEC.ER: .WORD   0               ;STORE SECTOR ADDRESS HERE
1346
1347                                  ;;*****************************************************************
1348
1349                                  .SBTTL  CONTROL PARAMETERS
1350
1351                                  ;;*****************************************************************
1352
1353  001274  137100                  ENDCON: .WORD   137100          ;6.25X10^8 WORDS (10) (1X10^10 BITS)
1354  001276  022500                          .WORD   22500           ;MSW
1355  001300  041100                  ENDSEK: .WORD   041100          ;1 X 10^6 SEEKS  (LSW)
1356  001302  000017                          .WORD   17              ;MSW
1357  001304  000001                  PASCNT: .WORD   1               ;NUMBER OF PASSES TO END OF TEST
1358  001306  000000                  BUFRSZ: .WORD   0               ;MAXIMUM DATA TRANFER SIZE IN WORDS
1359                                                                  ;(FILLED BY PROGRAM AT STARTUP OR BY OPERATOR
1360                                                                  ;DURING PARAMETER ENTRY DIALOG.)
1361  001310  000144                  MAXER:  .WORD   100.            ;MAXIMUM ERRORS - 100(10)
1362  001312  000005                  INTRVL: .WORD   5               ;LOW BYTE IS PERFORMANCE TYPEOUT INTERVAL
1363                                                                  ;COUNTER.  UPPER BYTE IS VALUE.
1364  001314  000004                  CMPLMT: .WORD   4               ;NUMBER OF COMPARE ERRORS TYPED OUT
1365  001316  000000                  SEKMOD: .WORD   0               ;IF EQ 1, END OF PASS DETERMINED BY SEEK COUNT
1366  001320  000003                  RATIO:  .WORD   3               ;READ/WRITE RATIO [RANGE 0 - 7]
1367                                                                  ;0 - 15/1       (READ/WRITE)
1368                                                                  ;1 - 7/1
1369                                                                  ;2 - 6/2
1370                                                                  ;3 - 5/3
1371                                                                  ;4 - 4/4
1372                                                                  ;5 - 3/5
1373                                                                  ;6 - 2/6
1374                                                                  ;7 - 1/7
1375  001322  000001                  AUTOCK: .WORD   1               ;IF EQ 1, DO A WRITE CHECK AFTER EACH ORDER
1376                                                                  ;IF EQ 0, SELECT WRITE CHECK ORDERS RANDOMLY
1377
1378                                  ;;*****************************************************************
1379
```

```
1380                                    .SBTTL  RP11E ADDRESS LIMIT VALUES
1381
1382                                    ;;*****************************************************************
1383
1384  001324  000000                    LC:     .WORD   0               ;MAX CYLINDER [RANGE 0 - 312 OR 625 (OCTAL)]
1385                                                                    ;THE PROGRAM WILL FILL THIS VALUE INITIALLY,
1386                                                                    ;THE VALUE WILL DEPEND ON THE TYPE OF DRIVE
1387                                                                    ;ON THE SYSTEM - RP02 OR RP03.
1388  001326  000000                    FC:     .WORD   0               ;MINIMUM CYLINDER [RANGE 0 - <= LC]
1389  001330  000023                    LT:     .WORD   23              ;MAX TRACK [RANGE 0 - 23 (OCTAL)]
1390  001332  000000                    FT:     .WORD   0               ;MIN TRACK [RANGE 0 TO <= LT]
1391  001334  000011                    LS:     .WORD   11              ;MAX SECTOR [RANGE 0 - 11 (OCTAL)]
1392  001336  000000                    FS:     .WORD   0               ;MIN SECTOR [RANGE 0 TO <- LS]
1393
1394                                    ;;*****************************************************************
1395
1396                                    .SBTTL  VALUES FOR FIRST OPERATION
1397
1398                                    ;;*****************************************************************
1399
1400  001340  000010                    BEGPAT: .WORD   10              ;STARTING PATTERN CODE [RANGE 1 - 17 (OCTAL)]
1401  001342  000005                    BEGCOD: .WORD   5               ;STARTING COMMAND CODE
1402                                                                    ;PRELOADED TO PERFORM A READ WITH IMPLIED SEEK
1403  001344  000400                    BEGSIZ: .WORD   400             ;STARTING RECORD SIZE [RANGE 4 - MAXMEM]
1404                                                                    ;NOTE:  THE SIZE MUST BE AT LEAST 4.
1405                                                                    ;IF THE SIZE IS GREATER THAN 1 SECTOR, THE
1406                                                                    ;SIZE MUST ALLOW FOR OVERLAPPING 4
1407                                                                    ;WORDS INTO THE LAST SECTOR USED.
```

```
                                        .SBTTL  ERROR POINTER TABLE

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

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


001346                                  $ERRTB: .

                                        ;ERROR 1

001346  027020                                  EM1                     ;RP11 DIDN'T RESPOND TO ADDRESSING
001350  030342                                  DH1
001352  032000                                  DT1
001354  032254                                  DF1

                                        ;ERROR 2

001356  027062                                  EM2                     ;INVALID RP11 COMMAND
001360  030350                                  DH2
001362  032002                                  DT2
001364  032260                                  DF2

                                        ;ERROR 3

001366  027107                                  EM3                     ;UNDEFINED INTERRUPT FROM RP11
001370  030365                                  DH3
001372  032006                                  DT3
001374  032260                                  DF2

                                        ;ERROR 4

001376  000000                                  0                       ;UNUSED
001400  000000                                  0
001402  000000                                  0
001404  000000                                  0

                                        ;ERROR 5

001406  000000                                  0                       ;UNUSED
001410  000000                                  0
001412  000000                                  0
001414  000000                                  0

                                        ;ERROR 6

001416  000000                                  0                       ;UNUSED
001420  000000                                  0
001422  000000                                  0
```

```
 1464  001424  000000                        0
 1465
 1466                                  ;ERROR 7
 1467
 1468  001426  000000                        0                       ;UNUSED
 1469  001430  000000                        0
 1470  001432  000000                        0
 1471  001434  000000                        0
 1472
 1473                                  ;ERROR 10
 1474
 1475  001436  027151                        EM10                    ;CAN'T MATCH DATA READ WITH A PATTTERN
 1476  001440  030402                        DH10
 1477  001442  032016                        DT10
 1478  001444  032264                        DF10
 1479
 1480                                  ;ERROR 11
 1481
 1482  001446  000000                        0                       ;SECOND LINE OF 'CAN'T MATCH DATA' MESSAGE
 1483  001450  000000                        0
 1484  001452  032012                        DT11
 1485  001454  032260                        DF2
 1486
 1487                                  ;ERROR 12
 1488
 1489  001456  027216                        EM12                    ;DATA COMPARE ERRORS
 1490  001460  030402                        DH10
 1491  001462  032016                        DT10
 1492  001464  032274                        DF12
 1493
 1494                                  ;ERROR 13
 1495
 1496  001466  000000                        0                       ;DATA COMPARE DETAIL LINE
 1497  001470  000000                        0
 1498  001472  032030                        DT13
 1499  001474  032304                        DF13
 1500
 1501                                  ;ERROR 14
 1502
 1503  001476  000000                        0                       ;DATA COMPARE ERROR SUMMARY LINE
 1504  001500  030561                        DH14
 1505  001502  032036                        DT14
 1506  001504  032310                        DF14
 1507
 1508                                  ;ERROR 15
 1509
 1510  001506  027242                        EM15                    ;'ERR' NOT SET BUT 'RPER' CONTENTS NOT ZERO
 1511  001510  030607                        DH15
 1512  001512  032040                        DT15
 1513  001514  032314                        DF15
 1514
 1515                                  ;ERROR 16
 1516
 1517  001516  027315                        EM16                    ;WORD COUNT NOT ZERO OR BUFFER ADDRESS WRONG
 1518  001520  030644                        DH16
 1519  001522  032050                        DT16
```

```
 1520  001524  032320 .                       DF16
 1521
 1522                                  ;ERROR 17
 1523
 1524  001526  027371                        EM17                    ;SOFTWARE TIMEOUT DURING POSITIONING
 1525  001530  030765                        DH17
 1526  001532  032066                        DT17
 1527  001534  032324                        DF17
 1528
 1529                                  ;ERROR 20
 1530
 1531  001536  027435                        EM20                    ;SOFTWARE TIMEOUT DURING I/O
 1532  001540  030765                        DH17
 1533  001542  032066                        DT17
 1534  001544  032324                        DF17
 1535
 1536                                  ;ERROR 21
 1537
 1538  001546  027471                        EM21                    ;DRIVE OFFLINE
 1539  001550  030765                        DH17
 1540  001552  032066                        DT17
 1541  001554  032324                        DF17
 1542
 1543                                  ;ERROR 22
 1544
 1545  001556  027507                        EM22                    ;DRIVE UNSAFE
 1546  001560  030765                        DH17
 1547  001562  032066                        DT17
 1548  001564  032324                        DF17
 1549
 1550                                  ;ERROR 23
 1551
 1552  001566  027524                        EM23                    ;SEEK INCOMPLETE
 1553  001570  031107                        DH23
 1554  001572  032114                        DT23
 1555  001574  032334                        DF23
 1556
 1557                                  ;ERROR 24
 1558
 1559  001576  027544                        EM24                    ;UNIBUS TRANSFER ERROR
 1560  001600  031227                        DH24
 1561  001602  032134                        DT24
 1562  001604  032344                        DF24
 1563
 1564                                  ;ERROR 25
 1565
 1566  001606  027572                        EM25                    ;CONTROLLER ERROR
 1567  001610  031274                        DH25
 1568  001612  032146                        DT25
 1569  001614  032314                        DF15
 1570
 1571                                  ;ERROR 26
 1572
 1573  001616  027613                        EM26                    ;NON-EXISTENT DISK ADDRESS
 1574  001620  031331                        DH26
 1575  001622  032156                        DT26
```

```
1576  001624  032350                        DF26
1577
1578                                  ;ERROR 27
1579
1580  001626  027645                        EM27                    ;DATA ERROR
1581  001630  031406                        DH27
1582  001632  032172                        DT27
1583  001634  032354                        DF27
1584
1585                                  ;ERROR 30
1586
1587  001636  000000                        0                       ;SOFT ERROR REPORT LINE
1588  001640  031560                        DH30
1589  001642  000000                        0
1590  001644  000000                        0
1591
1592                                  ;ERROR 31
1593
1594  001646  000000                        0                       ;HARD ERROR REPORT LINE
1595  001650  031573                        DH31
1596  001652  000000                        0
1597  001654  000000                        0
1598
1599                                  ;ERROR 32
1600
1601  001656  027660                        EM32                    ;WRITE CHECK ERROR (NO DATA ERROR)
1602  001660  031406                        DH27
1603  001662  032172                        DT27
1604  001664  032354                        DF27
1605
1606                                  ;ERROR 33
1607
1608  001666  027722                        EM33                    ;WRITE CHECK ERROR (DATA ERROR BITS SET)
1609  001670  031406                        DH27
1610  001672  032172                        DT27
1611  001674  032354                        DF27
1612
1613                                  ;ERROR 34
1614
1615  001676  027772                        EM34                    ;HEADER NOT FOUND
1616  001700  031606                        DH34
1617  001702  032226                        DT34
1618  001704  032364                        DF34
1619
1620                                  ;ERROR 35
1621
1622  001706  030013                        EM35                    ;FORMAT ERROR WHILE SEARCHING FOR INITIAL SECTOR
1623  001710  031406                        DH27
1624  001712  032172                        DT27
1625  001714  032354                        DF27
1626
1627                                  ;ERROR 36
1628
1629  001716  030073                        EM36                    ; HEADER NOT FOUND, CYLINDER IN HEADER
1630                                                                ;DOES NOT COMPARE
1631  001720  031606                        DH34
```

```
1632  001722  032226                          DT34
1633  001724  032364                          DF34
1634
1635                                    ;ERROR 37
1636
1637  001726  030161                          EM37                    ;DIFFERENT ERROR DURING RETRY
1638  001730  030765                          DH17
1639  001732  032066                          DT17
1640  001734  032324                          DF17
1641
1642                                    ;ERROR 40
1643
1644  001736  000000                          0                       ;UNUSED
1645  001740  000000                          0
1646  001742  000000                          0
1647  001744  000000                          0
1648
1649                                    ;ERROR 41
1650
1651  001746  000000                          0                       ;RETRY SUCESSFUL
1652  001750  031735                          DH41
1653  001752  000000                          0
1654  001754  000000                          0
1655
1656                                    ;ERROR 42
1657
1658  001756  000000                          0                       ;RETRY UNSUCESSFUL
1659  001760  031755                          DH42
1660  001762  000000                          0
1661  001764  000000                          0
1662
1663                                    ;ERROR 43
1664
1665  001766  030216                          EM43                    ;ERROR SIGNALED BUT NO ERROR BITS SET
1666  001770  030765                          DH17
1667  001772  032066                          DT17
1668  001774  032324                          DF17
1669
1670                                    ;ERROR 44
1671
1672  001776  030257                          EM44                    ;WRITE PROTECT VIOLATION
1673  002000  030607                          DH15
1674  002002  032040                          DT15
1675  002004  032314                          DF15
1676
1677                                    ;ERROR 45
1678
1679  002006  030312                          EM45                    ;FORMAT ERROR DURING I/O
1680  002010  031406                          DH27
1681  002012  032172                          DT27
1682  002014  032354                          DF27
1683
1684                                    ;;**********************************************************************
1685
1686                                    .SBTTL  SETUP AND 'NITIALIZATION ROUTINE
1687
```

```
1688                                    ;       START ADDRESS = 200
1689                                    ;       START ADDRESS WITH PARAMETER CHANGES = 204
1690
1691                                    ;;**********************************************************************
1692
1693  002016  005037  001262           START:  CLR     CHNGPM          ;CLEAR 'START FROM 204' INDICATOR
1694  002022  000403                           BR      BEGIN           ;START THE PROGRAM
1695  002024  012737  177777  001262   START1: MOV     #-1,CHNGPM      ;SET 'CHANGE PARAMETER' INDICATOR
1696  002032  000005                   BEGIN:  RESET                   ;CLEAR EVERYTHING
1697                                    .SBTTL  INITIALIZE THE COMMON TAGS
1698                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1699  002034  012706  001100                   MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1700  002040  005026                           CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1701  002042  022706  001140                   CMP     #SWR,R6 ;;DONE?
1702  002046  001374                           BNE     .-6             ;;LOOP BACK IF NO
1703  002050  012706  001100                   MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1704                                    .;INITIALIZE A FEW VECTORS
1705  002054  012737  015614  000030           MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1706  002062  012737  000340  000032           MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1707  002070  012737  021010  000034           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1708  002076  012737  000340  000036           MOV     #340,@#TRAPVEC+2;LEVEL 7
1709  002104  012737  176543  020610           MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
1710  002112  012737  123456  020612           MOV     #123456,$LONUM  ;;BOTH HIGH AND LOW WORDS
1711                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1712                                    ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1713  002120  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1714  002124  012737  002160  000004           MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1715  002132  012737  177570  001140           MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1716  002140  012737  177570  001142           MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1717  002146  012777  177777  176764           CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1718  002154  001012                           BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1719                                                                   ;;AND  THE HARDWARE SWR IS NOT   -1
1720  002156  000403                           BR      65$             ;;BRANCH IF NO TIMEOUT
1721  002160  012716  002166           64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1722  002164  000002                           RTI
1723  002166  012737  000176  001140   65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1724  002174  012737  000174  001142           MOV     #DISPREG,DISPLAY
1725  002202  012637  000004           66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1726
1727  002206  012737  000240  000032           MOV     #240,@#EMTVEC+2 ;CHANGE EMT PRIORITY TO 5
1728  002214  012737  000240  000036           MOV     #240,@#TRAPVEC+2        ;CHANGE TRAP PRIORITY TO 5
1729  002222  104401  033634           1$:     TYPE    ,TITLE          ;TYPE THE PROGRAM NAME AND MAINDEC NUMBER
1730  002226  012737  001213  002224           MOV     #$CRLF,1$+2     ;CHANGE 'TITLE' TO CR-LF
1731  002234  005037  177776                   CLR     PS              ;RESET PROCESSOR PRIORITY
1732  002240  004737  016746                   JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
1733                                    .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1734  002244  005737  000042           TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1735  002250  001006                   BNE     67$             ;;BRANCH IF YES
1736  002252  023727  001140  000176   CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1737  002260  001005                   BNE     68$             ;;BRANCH IF NO
1738  002262  104406                   GTSWR                   ;;GET SOFT-SWR SETTINGS
1739  002264  000403                   BR      68$
1740  002266  112737  000001  001134   67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1741  002274                           68$:
1742  002274  005037  001250                   CLR     STATIN          ;CLEAR PERFORMANCE SUMMARY INDICATOR
1743  002300  105037  001157                   CLRB    $TPFLG          ;SET TTY AVAILABILITY FLAG
```

```
1744  002304  012705  025450            MOV     #ASNLST,R5      ;START OF AREA TO CLEAR
1745  002310  005025                2$:  CLR     (R5)+           ;CLEAR
1746  002312  022705  025726            CMP     #BLKADR,R5      ;LOOK FOR END OF CLEAR AREA
1747  002316  001374                    BNE     2$              ;BR IF NOT FINISHED
1748  002320  013737  001240  010134    MOV     HZ,SIXTEE       ;1/60 TH OR 1/50 TH SECOND COUNTER VALUE
1749  002326  005437  010134            NEG     SIXTEE          ;CONVERT TO 2'S COMP
1750  002332  012737  030060  010120    MOV     #30060,HOUR     ;ASCII '00' TO HOUR COUNTER
1751  002340  012737  030060  010124    MOV     #30060,MINUTE   ;ASCII '00' TO MINUTES COUNTER
1752  002346  012737  030060  010130    MOV     #30060,SECOND   ;ASCII '00' TO SECONDS COUNTER
1753  002354  105037  001313            CLRB    INTRVL+1        ;CLEAR INTERVAL COUNTER
1754  002360  005037  177776            CLR     PSW             ;CLEAR PROCESSOR STATUS
1755  002364  005037  001252            CLR     PACK            ;'R' OR 'W' COMMAND INDICATOR
1756
1757                                ;ROUTINE TO DETERMINE BUFFER AREA SIZE
1758
1759  002370  005227  177777    SIZMEM: INC     #-1             ;FIRST START ?
1760  002374  001013                    BNE     1$              ;BR IF NOT
1761  002376  004737  033716            JSR     PC,$SIZE        ;SEE HOW MUCH MEMORY ON SYSTEM
1762  002402  012737  000137  002370    MOV     #137,SIZMEM     ;CLEAR ENTRY TO '$SIZE'
1763  002410  012737  002424  002372    MOV     #1$,SIZMEM+2    ;SAME
1764  002416  013737  034012  001244    MOV     $LSTAD,LSTAD    ;SAVE THE HIGH MEMORY ADDRESS
1765  002424  012737  000001  025604 1$: MOV     #1,BUFTBL       ;LOAD NUMBER OF BUFFERS
1766  002432  012737  033634  025606    MOV     #ENDPGM,BUFTBL+2  ;STARTING ADDRESS OF BUFFER
1767  002440  013737  001244  025610    MOV     LSTAD,BUFTBL+4  ;LAST ADDR TO BUFFER ALLOCATION TABLE
1768  002446  162737  033634  025610    SUB     #ENDPGM,BUFTBL+4  ;SUBTRACT PROGRAM SPACE
1769  002454  000241                    CLC                     ;CLEAR THE 'C' BIT
1770  002456  006037  025610            ROR     BUFTBL+4        ;CONVERT TO WORD COUNT
1771  002462  022737  037776  001244    CMP     #37776,LSTAD    ;ONLY 8K ON THE SYSTEM ?
1772  002470  103003                    BHIS    2$              ;BR IF 8K OR LESS
1773  002472  105737  000041            TSTB    41              ;SEE WHO LOADED THE PROGRAM
1774  002476  001004                    BNE     3$              ;BR IF LOADED BY 'XXDP'
1775  002500  162737  000144  025610 2$: SUB     #100.,BUFTBL+4  ;SUBTRACT 'ABS' LOADER SIZE
1776  002506  000403                    BR      4$              ;CONTINUE WITH BUFFER SPACE SETUP
1777  002510  162737  002734  025610 3$: SUB     #1500.,BUFTBL+4  ;SUBTRACT 'XXDP' LOADER SIZE
1778  002516  005737  001306         4$: TST     BUFRSZ          ;VALUE IN 'BUFRSZ' ?
1779  002522  001012                    BNE     5$              ;BR IF VALUE IS
1780  002524  012737  005400  001306    MOV     #2816.,BUFRSZ   ;ASSUME FULL TRACK + 1 SEC MAXIMUM
1781  002532  023737  001306  025610    CMP     BUFRSZ,BUFTBL+4 ;IS THAT TOO LARGE ?
1782  002540  101403                    BLOS    5$              ;BR IF NOT
1783  002542  013737  025610  001306    MOV     BUFTBL+4,BUFRSZ ;USE MAX AVAIL MEMORY AS MAX BUFFER SIZE
1784  002550  005737  001262         5$: TST     CHNGPM          ;CHANGE PARAMETERS ?
1785  002554  001455                    BEQ     CKADR           ;BR IF NOT
1786
1787                                ;PARAMETER ENTRY ROUTINE
1788
1789  002556  012737  002662  001242 ENTPAR: MOV     #ENTPR1,CNTRLC  ;CONTROL C RETURN ADDRESS
1790  002564  104401  033364        ENTPR:  TYPE    ,ASKPAR         ;'ENTER PARAMETERS'
1791  002570  012704  033266                MOV     #PARLST,R4      ;ENTRY PARAMETER LIST TO R4
1792  002574  012437  002612         1$:    MOV     (R4)+,4$        ;ADDRESS OF PARAMETER NAME
1793  002600  001435                        BEQ     ENTPR2          ;BR IF AT END OF TABLE
1794  002602  000402                        BR      3$              ;GET AROUND THE CR-LF
1795  002604  104401  001213         2$:    TYPE    ,$CRLF          ;CR-LF
1796  002610  104401                 3$:    TYPE                    ;TYPE THE PARAMETER NAME
1797  002612  000000                 4$:    .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
1798  002614  011405                        MOV     (R4),R5         ;ADDRESS OF PARAMETER LOCATION
1799  002616  011546                        MOV     (R5),-(SP)      ;CONTENTS OF PARAMETER LOCATION
```

```
1800  002620  104402                          TYPOC                   ;TYPE THE CURRENT VALUE
1801  002622  104401  032471                  TYPE    ,SPACE1         ;1 SPACE
1802  002626  104411                          RDLIN                   ;READ THE KEYBOARD
1803  002630  012601                          MOV     (SP)+,R1        ;BUFFER ADDRESS
1804  002632  004037  012010                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
1805  002636  002604                          2$                      ;ILLEGAL INPUT
1806  002640  002650                          5$                      ;TERMINATED WITH A 5$
1807  002642  002656                          7$                      ;TERMINATED WITH A '.'
1808  002644  002652                          6$                      ;NO ENTRY - TERMINATED WITH A 5$
1809  002646  002674                          ENTPR2                  ;NO ENTRY - TERMINATED WITH A '.'
1810  002650  010215                  5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
1811  002652  005724                  6$:     TST     (R4)+           ;INCREMENT THE PARAMETER POINTER
1812  002654  000747                          BR      1$              ;GET MORE PARAMETERS
1813  002656  010215                  7$:     MOV     R2,(R5)         ;NEW PARAMETER VALUE
1814  002660  000405                          BR      ENTPR2          ;CONTINUE
1815  002662  012706  001100          ENTPR1: MOV     #STACK,SP       ;RESET THE STACK POINTER
1816  002666  005037  177776                  CLR     PSW             ;CLEAR THE STATUS WORD
1817  002672  000734                          BR      ENTPR           ;START WITH PARAMETER ENTRY AGAIN
1818  002674  005077  176244          ENTPR2: CLR     @$TKS           ;CLEAR THE KEYBOARD INTERRUPT
1819  002700  005037  177776                  CLR     PSW             ;CLEAR PROCESSOR STATUS
1820  002704  012706  001100                  MOV     #STACK,SP       ;RESTORE STACK POINTER FOR RESTART
1821
1822                                          ;CHECK THE PARAMETERS FOR VALIDITY
1823
1824  002710  012737  002732  000004  CKADR:  MOV     #1$,ERRVEC      ;CHANGE TIMEOUT RETURN
1825  002716  005777  176274                  TST     @RPADR          ;SEE IF RP11 RESPONDS
1826  002722  012737  000006  000004          MOV     #ERRVEC+2,ERRVEC  ;RESET THE TIMEOUT VECTOR
1827  002730  000411                          BR      2$              ;CONTINUE WITH CHECKING
1828  002732  012737  000006  000004  1$:     MOV     #ERRVEC+2,ERRVEC  ;RESET TIMEOUT VECTOR
1829  002740  104001                          ERROR   1               ;REPORT RP11 DIDN'T RESPOND
1830  002742  013700  000042                  MOV     @#42,R0         ;'CHAIN MODE ' OR 'ACT11' ?
1831  002746  001703                          BEQ     ENTPAR          ;IF NOT - GO TO ENTER PARAMETER ROUTINE
1832  002750  000137  003634                  JMP     $ENDAD          ;RETURN TO MONITOR
1833  002754  004737  021304          2$:     JSR     PC,RPINIT       ;INITIALIZE THE RP11 HANDLER
1834  002760  012737  177777  021122          MOV     #-1,SAVEFG      ;SET THE 'SAVE REGISTER' FLAG
1835  002766  012737  000312  001256          MOV     #202.,MAXCYL    ;START CYLINDER LIMIT AT RP02 LIMIT
1836  002774  005737  021112                  TST     DRVTYP          ;SEE WHICH DRIVES (RP02 OR RP03) ARE
1837                                                                  ;ON SYSTEM
1838  003000  001403                          BEQ     3$              ;BR IF RP02'S
1839  003002  012737  000625  001256          MOV     #405.,MAXCYL    ;CYLINDER LIMIT FOR RP03'S
1840  003010  005737  001324          3$:     TST     LC              ;VALUE ENTERED IN 'LC' ?
1841  003014  001003                          BNE     4$              ;BR IF ONE WAS
1842  003016  013737  001256  001324          MOV     MAXCYL,LC       ;USE MAXIMUM VALUE
1843  003024  005037  001262          4$:     CLR     CHNGPM          ;CLEAR CHANGE PARAMETERS INDICATOR
1844  003030  023737  001306  025610          CMP     BUFRSZ,BUFTBL+4 ;REQUESTED TRANSFER SIZE TOO LARGE ?
1845  003036  101004                          BHI     5$              ;BR IF REQUESTED MAXIMUM TOO LARGE
1846  003040  022737  000004  001306          CMP     #4,BUFRSZ       ;SEE IF MAXIMUM TRANSFER SIZE TOO SMALL
1847  003046  101404                          BLOS    6$              ;BR IF NOT TOO SMALL
1848  003050  012746  033471          5$:     MOV     #PAR1,-(SP)     ;ADDR OF PARAMETER NAME
1849  003054  000137  003256                  JMP     BADPAR          ;REPORT THE PARAMETER ERROR
1850  003060  023737  001306  001344  6$:     CMP     BUFRSZ,BEGSIZ   ;IS MAX RECORD LENGTH SMALLER THAN 1 SECTOR ?
1851  003066  103003                          BHIS    CKPAR           ;BR IF NOT
1852  003070  013737  001306  001344          MOV     BUFRSZ,BEGSIZ   ;USE MAX RECORD SIZE AS STARTING SIZE
1853  003076  023737  001256  001324  CKPAR:  CMP     MAXCYL,LC       ;MAXIMUM CYLINDER ADDR TOO LARGE ?
1854  003104  103004                          BHIS    1$              ;BR IF NOT
1855  003106  012746  033521                  MOV     #PAR4,-(SP)     ;ADDR OF PARAMETER NAME
```

```
1856  003112  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1857  003116  023737  001324  001326  1$:       CMP     LC,FC           ;MAX CYLINDER LARGER THAN MIN CYLINDER
1858  003124  103004                            BHIS    2$              ;BR IF IT IS
1859  003126  012746  033525                    MOV     #PAR5,-(SP)     ;ADDR OF PARAMETER NAME
1860  003132  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1861  003136  022737  000023  001330  2$:       CMP     #19.,LT         ;MAXIMUM TRACK ADDRESS TOO LARGE ?
1862  003144  103004                            BHIS    3$              ;BR IF NOT
1863  003146  012746  033531                    MOV     #PAR6,-(SP)     ;ADDRESS OF PARAMETER NAME
1864  003152  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1865  003156  023737  001330  001332  3$:       CMP     LT,FT           ;SEE IF MAX TRACK EQ OR LARGER THAN MIN TRACK
1866  003164  103004                            BHIS    4$              ;BR IF IT IS
1867  003166  012746  033535                    MOV     #PAR7,-(SP)     ;ADDRESS OF PARAMETER NAME
1868  003172  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1869  003176  022737  000011  001334  4$:       CMP     #9.,LS          ;SEE IF MAX SECTOR TOO LARGE ?
1870  003204  103004                            BHIS    5$              ;BR IF MAX SECTOR OK
1871  003206  012746  033541                    MOV     #PAR8,-(SP)     ;ADDRESS OF PARAMETER NAME
1872  003212  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1873  003216  023737  001334  001336  5$:       CMP     LS,FS           ;MINIMUM SECTOR ADDRESS TOO LARGE
1874  003224  103004                            BHIS    6$              ;BR IF NOT
1875  003226  012746  033545                    MOV     #PAR9,-(SP)     ;ADDRESS OF PARAMETER NAME
1876  003232  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1877  003236  022737  000007  001320  6$:       CMP     #7,RATIO        ;SEE IF R/W RATIO TOO LARGE
1878  003244  103016                            BHIS    SETVEC          ;BR IF NOT
1879  003246  012746  033577                    MOV     #PAR14,-(SP)    ;ADDRESS OF PARAMETER NAME
1880  003252  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
1881
1882                                    ;REPORT THE BAD PARAMETER
1883
1884  003256  012637  003270            BADPAR: MOV     (SP)+,1$        ;MOVE PARAMETER NAME
1885  003262  104401  033411                    TYPE    ,NGPAR          ;'BAD PARAMETER'
1886  003266  104401                            TYPE
1887  003270  000000                    1$:     .WORD   0               ;PARAMETER NAME .
1888  003272  104401  033441                    TYPE    ,REPAR          ;'REENTER PARAMETERS
1889  003276  000137  002556                    JMP     ENTPAR          ;GO THROUGH PARAMETER ENTRY ROUTINE
1890
1891                                    ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
1892                                    ;       PROGRAM WILL USE
1893
1894  003302  005227  177777            SETVEC: INC     #-1             ;FIRST START ?
1895  003306  001404                            BEQ     9$              ;BR IF IT IS
1896  003310  032777  000200  175622            BIT     #SW07,@SWR      ;SWITCH 6 SET ?
1897  003316  001064                            BNE     8$              ;BR IF SET
1898  003320  005004                    9$:     CLR     R4              ;DRIVE TABLE POINTER
1899  003322  012703  000010                    MOV     #8.,R3          ;ITERATION COUNTER
1900  003326  104401  001213                    TYPE    ,$CRLF          ;CR-LF
1901  003332  104401  032615                    TYPE    ,SYSTAT         ;TYPE STATUS HEADING
1902  003336  010446                    1$:     MOV     R4,-(SP)        ;DRIVE NUMBER FOR TYPEOUT
1903  003340  104403                            TYPOS                   ;TYPE THE DRIVE NUMBER .
1904  003342     001     000                    .BYTE   1,0
1905  003344  104401  032466                    TYPE    ,SPACE4         ;SPACES
1906  003350  105764  021102                    TSTB    DRVSTA(R4)      ;SEE WHAT DRIVE'S STATUS IS
1907  003354  001405                            BEQ     2$              ;BR IF OFFLINE OR NOT PRESENT
1908  003356  100413                            BMI     3$              ;BR IF UNSAFE
1909  003360  012737  032525  003442            MOV     #UNTON,6$       ;MESSAGE ADDRESS
1910  003366  000412                            BR      4$              ;DETERMINE DRIVE TYPE MESSAGE
1911  003370  012737  032514  003442  2$:       MOV     #UNTOFF,6$      ;OFFLINE MESSAGE ADDRESS
```

```
                        GET VALUE FOR SOFTWARE SWITCH REGISTER                                          SEQ 0036

 1912  003376  012737  032471  003452          MOV     #SPACE1,7$      ;DUMMY MESSAGE
 1913  003404  000415                          BR      5$              ;PRINT IT
 1914  003406  012737  032605  003442  3$:     MOV     #NOTSAF,6$      ;UNIT UNSAFE
 1915  003414  012737  032473  003452  4$:     MOV     #RP02,7$        ;ASSUME RP02
 1916  003422  136437  021274  021112          BITB    ATABIT(R4),DRVTYP  ;SEE WHICH DRIVE TYPE
 1917  003430  001403                          BEQ     5$              ;BR IF RP02
 1918  003432  012737  032500  003452          MOV     #RP03,7$        ;'RP03'
 1919  003440  104401                  5$:     TYPE                    ;TYPE THE IDENTIFYING MESSAGE
 1920  003442  000000                  6$:     .WORD   0               ;MESSAGE ADDRESS HERE
 1921  003444  104401  032466                  TYPE    ,SPACE4         ;SPACES
 1922  003450  104401                          TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
 1923  003452  000000                  7$:     .WORD   0               ;DRIVE TYPE MESSAGE GOES HERE
 1924  003454  104401  001213                  TYPE    ,$CRLF          ;CR-LF
 1925  003460  005303                          DEC     R3              ;DECREMENT THE ITERATION COUNTER
 1926  003462  001402                          BEQ     8$              ;BR IF END
 1927  003464  005204                          INC     R4              ;INCREMENT TABLE POINTER
 1928  003466  000723                          BR      1$              ;CONTINUE
 1929  003470  104401  033202          8$:     TYPE    ,INTDON         ;TYPE 'INITIALIZE COMPLETE'
 1930  003474  004737  006752                  JSR     PC,CKCLK        ;START CLOCK IF ONE ON SYSTEM
 1931  003500  012737  000200  177776          MOV     #PR4,PSW        ;SET PRIORITY TO KEYBOARD LEVEL
 1932  003506  004737  016746                  JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD INTERRUPT HANDLER
 1933  003512  012737  006630  001242          MOV     #DORTI,CNTRLC   ;ADDRESS IF 'CONTROL C'
 1934  003520  012737  010136  000060          MOV     #KSR,TKVEC      ;CHANGE KEYBOARD VECTOR ADDRESS
 1935  003526  005037  177776                  CLR     PSW             ;LOWER PRIORITY
 1936
 1937                                  ;SET UP IF 'XXDP' OR 'ACT11' OPERATION
 1938
 1939  003532  005737  000042          MONTR:  TST     42              ;'XXDP' CHAIN MODE OR 'ACT11' AUTO ACCEPT
 1940  003536  001417                          BEQ     2$              ;BR IF NEITHER
 1941  003540  012703  000010                  MOV     #8.,R3          ;DRIVE COUNTER
 1942  003544  005004                          CLR     R4              ;DRIVE POINTER
 1943  003546  105737  000041                  TSTB    41              ;SEE WHICH - 'XXDP' OR 'ACT11' ?
 1944  003552  001407                          BEQ     1$              ;BR IF 'ACT11' CONTROL
 1945  003554  122737  000007  000041          CMPB    #7,41           ;IS RP02 OR RP03 THE LOADING DEVICE ?
 1946  003562  001003                          BNE     1$              ;BR IF NOT
 1947  003564  005204                          INC     R4              ;SETUP TO BYPASS DRIVE 0
 1948  003566  012703  000007                  MOV     #7,R3           ;DRIVES TO TRY TO ASSIGN
 1949  003572  004737  010472          1$:     JSR     PC,ASGN2        ;ASSIGN ALL AVAILABLE DRIVES (EXCEPT DRIVE
 1950                                                                  ;0 IF LOADED FROM AN RP02 OR RP03)
 1951  003576  000137  004046          2$:     JMP     MAIN1           ;START THE PROGRAM
 1952
 1953                                  ;'XXDP' OR 'ACT11' END OF TEST ROUTINE
 1954
 1955  003602  013700  000042          $GET42: MOV     42,R0           ;MONITOR ADDRESS
 1956  003606  001002                          BNE     1$              ;BR IF MONITOR ADDRESS
 1957  003610  000137  004046                  JMP     MAIN1           ;NONE, CONTINUE
 1958  003614  022700  003634          1$:     CMP     #$ENDAD,R0      ;IS MONITOR 'ACT11'
 1959  003620  001005                          BNE     $ENDAD          ;NO, BRANCH
 1960  003622  022760  177777  000002          CMP     #-1,2(R0)       ;YES, IF THIS LAST PASS
 1961  003630  001005                          BNE     $DOAGN          ;NO, MAKE ANOTHER PASS
 1962  003632  000005                          RESET                   ;CLEAR EVERYTHING
 1963  003634  004710                  $ENDAD: JSR     PC,(R0)         ;GO TO THE MONITOR
 1964  003636  000240                          NOP                     ;SAVE ROOM
 1965  003640  000240                          NOP                     ;FOR
 1966  003642  000240                          NOP                     ;ACT11
 1967  003644  000137  002016          $DOAGN: JMP     START           ;START AGAIN
```

```
 1968
 1969                                   ;;*****************************************************************
 1970
 1971                                   .SBTTL  MAIN PROGRAM
 1972
 1973                                   ;;*****************************************************************
 1974
 1975  003650  005737  001250          MAIN:   TST     STATIN          ;TYPE ERROR RATE STATISTICS ?
 1976  003654  001410                          BEQ     MAIN0           ;BR IF NOT
 1977  003656  032777  000100  175254          BIT     #SW06,@SWR      ;CHECK SWITCH 6
 1978  003664  001004                          BNE     MAIN0           ;BR IF SET
 1979  003666  005037  001250                  CLR     STATIN          ;CLEAR THE INDICATOR
 1980  003672  004737  007116                  JSR     PC,STATPR       ;TYPE THE STATISTICS
 1981
 1982                                   ;LOOK FOR DRIVES TO BE DEASSIGNED
 1983
 1984  003676  012703  000010          MAIN0:  MOV     #8.,R3          ;UNIT COUNTER
 1985  003702  012705  025474                  MOV     #DUNIT,R5       ;ADDRESS OF 'DROP UNIT' TABLE
 1986  003706  005715                  1$:     TST     (R5)            ;SEE IF ENTRY AT PRESENT POSITION
 1987  003710  001011                          BNE     3$              ;BR IF THERE IS ONE
 1988  003712  062705  000002          2$:     ADD     #2,R5           ;INCREMENT TO NEXT TABLE POSITION
 1989  003716  005303                          DEC     R3              ;DECREMENT UNIT COUNTER
 1990  003720  001372                          BNE     1$              ;BR IF MORE TO CHECK
 1991  003722  005737  025450                  TST     ASNLST          ;ANY DRIVES STILL RUNNING ?
 1992  003726  001047                          BNE     MAIN1           ;BR IF ANY ARE - CHECK ASSIGN QUEUE
 1993  003730  000137  003602                  JMP     $GET42          ;CHECK FOR MONITOR RETURN
 1994  003734  012701  025540          3$:     MOV     #AVAIL,R1       ;ADDRESS OF 'AVAILABLE' UNITS TABLE
 1995  003740  005711                  4$:     TST     (R1)            ;SEE IF AT END OF TABLE
 1996  003742  001405                          BEQ     5$              ;BR IF AT END: GO CHECK 'WAIT' TABLE
 1997  003744  021115                          CMP     (R1),(R5)       ;IS UNIT IN 'AVAIL' THE ONE TO BE DROPPED
 1998  003746  001414                          BEQ     7$              ;BR IF YES
 1999  003750  062701  000002                  ADD     #2,R1           ;INCREMENT 'AVAIL' TABLE ADDRESS
 2000  003754  000771                          BR      4$              ;CONTINUE LOOKING
 2001  003756  012701  025562          5$:     MOV     #WAIT,R1        ;MOVE THE ADDRESS OF THE BUFFER WAIT TABLE
 2002  003762  005711                  6$:     TST     (R1)            ;AT THE END OF THE 'WAIT' TABLE ?
 2003  003764  001752                          BEQ     2$              ;BR IF YES: SEE IF ANY MORE 'DROP' REQUESTS
 2004  003766  021115                          CMP     (R1),(R5)       ;UNIT IN THE 'WAIT' TABLE ?
 2005  003770  001403                          BEQ     7$              ;BR IF IT IS
 2006  003772  062701  000002                  ADD     #2,R1           ;INCREMENT 'WAIT' TABLE ADDRESS
 2007  003776  000771                          BR      6$              ;CONTINUE LOOK THROUGH THE 'WAIT' TABLE
 2008  004000  011100                  7$:     MOV     (R1),R0         ;PUT THE UNIT'S BLOCK ADDRESS IN R0
 2009  004002  104401  001213                  TYPE    ,$CRLF          ;CR-LF
 2010  004006  004737  007624                  JSR     PC,$TIME        ;TYPE THE TIME
 2011  004012  104401  032505                  TYPE    ,UNITMSG        ;'DRIVE'
 2012  004016  111046                          MOVB    (R0),-(SP)      ;DRIVE NUMBER
 2013  004020  104403                          TYPOS                   ;TYPE IT
 2014  004022     001     000                  .BYTE   1,0
 2015  004024  104401  033057                  TYPE    ,DEASSG         ;'DEASSIGNED'
 2016  004030  004737  007230                  JSR     PC,TYPEST       ;TYPE THE DRIVE'S PERFORMANCE SUMMARY
 2017  004034  005015                          CLR     (R5)            ;CLEAR THE 'DROP UNIT' TABLE ENTRY
 2018  004036  005011                          CLR     (R1)            ;REMOVE THE UNIT FROM THE 'AVAIL' OR 'WAIT' TABLE
 2019  004040  004737  006640                  JSR     PC,CMPRES       ;COMPRESS THE RESPECTIVE TABLE
 2020  004044  000722                          BR      2$              ;SEE IF ANY MORE UNITS
 2021
 2022                                   ;LOOK FOR DRIVES TO BE ASSIGNED
 2023
```

```
2024  004046  012703  000010          MAIN1:  MOV     #8.,R3          ;UNIT COUNT
2025  004052  005002                          CLR     R2              ;'AVAIL' INDEX
2026  004054  005004                          CLR     R4              ;ASSIGN LIST INDEX
2027  004056  005005                          CLR     R5              ;NEW UNIT INDEX
2028  004060  005765  025516          1$:     TST     NEWUNT(R5)      ;NEW UNIT IN THIS POSITION
2029  004064  001006                          BNE     3$              ;BR IF THERE IS
2030  004066  062705  000002          2$:     ADD     #2,R5           ;INCREMENT R5
2031  004072  005204                          INC     R4              ;INCREMENT ASSIGN INDEX
2032  004074  005303                          DEC     R3              ;DECREMENT UNIT COUNT
2033  004076  001370                          BNE     1$              ;BR IF MORE UNITS
2034  004100  000472                          BR      MAIN2           ;START OPERATIONS FOR THE AVAILABLE UNITS
2035  004102  104401  001213          3$:     TYPE    ,$CRLF          ;CR-LF
2036  004106  004737  007624                  JSR     PC,$TIME        ;TYPE THE TIME
2037  004112  104401  032505                  TYPE    ,UNTMSG         ;'DRIVE'
2038  004116  010446                          MOV     R4,-(SP)        ;DRIVE NUMBER
2039  004120  104403                          TYPOS                   ;TYPE IT
2040  004122     001     000                  .BYTE   1,0
2041  004124  104401  033121                  TYPE    ,ASGND          ;'ASSIGNED'
2042  004130  005762  025540          4$:     TST     AVAIL(R2)       ;AT END OF AVAILABLE TABLE
2043  004134  001403                          BEQ     5$              ;BR IF YES
2044  004136  062702  000002                  ADD     #2,R2           ;INCREMENT AVAILABLE TABLE INDEX
2045  004142  000772                          BR      4$              ;CONTINUE LOOKING FOR END OF TABLE
2046  004144  016562  025516  025540  5$:     MOV     NEWUNT(R5),AVAIL(R2)  ;MOVE ADDR OF UNIL INTO AVAIL LST
2047  004152  005065  025516                  CLR     NEWUNT(R5)      ;TAKE UNIT OUT OF NEW UNIT TABLE
2048  004156  156437  021274  025450          BISB    ATABIT(R4),ASNLST  ;SET UNIT ASSIGNED INDICATOR
2049  004164  016200  025540                  MOV     AVAIL(R2),R0    ;PUT STARTING ADDRESS OF BLOCK IN R0
2050  004170  113760  001336  000012          MOVB    FS,$SEC(R0)  ;INITIAL SECTOR VALUE
2051  004176  113760  001332  000013          MOVB    FT,$TRK(R0)  ;INITIAL TRACK VALUE
2052  004204  013760  001326  000010          MOV     FC,$CYL(R0)  ;INITIAL CYLINDER VALUE
2053  004212  113760  001342  000002          MOVB    BEGCOD,$COMND(R0)  ;INITIAL COMMAND CODE
2054  004220  113760  001340  000026          MOVB    BEGPAT,$PATTC(R0)  ;PATTERN CODE
2055  004226  106360  000026                  ASLB    $PATTC(R0)      ;CONVERT CODE TO A TABLE INDEX
2056  004232  013760  001344  000020          MOV     BEGSIZ,$WRDL(R0)  ;BEGINNING RECORD SIZE
2057  004240  013760  001344  000004          MOV     BEGSIZ,$WRDM(R0)  ;VALUE FOR DATA TRANSFER
2058  004246  005460  000004                  NEG     $WRDM(R0)       ;MAKE IT INTO 2'S COMPLEMENT
2059  004252  013760  001254  000064          MOV     FAIRNS,$FAIR(R0)  ;LOAD FAIRNESS COUNT
2060  004260  062702  000002                  ADD     #2,R2           ;INCREMENT 'AVAILABLE' TABLE POINTER
2061  004264  000700                          BR      2$              ;LOOK FOR MORE UNITS
2062
2063                                  ;GET PARAMETERS, BUFFER SPACE, AND START ORDERS FOR DRIVES IN
2064                                  ;       THE 'AVAILABLE' QUEUE
2065
2066  004266  005737  025562          MAIN2:  TST     WAIT            ;OUTSTANDING BUFFER REQUESTS
2067  004272  001061                          BNE     MAIN3           ;BR IF THERE ARE
2068  004274  005002                          CLR     R2              ;CLEAR 'AVAIL' TABLE POINTER
2069  004276  005762  025540          1$:     TST     AVAIL(R2)       ;ANY UNITS WAITING FOR PARAMETERS
2070  004302  001507                          BEQ     IDLE            ;BRANCH IF NONE
2071  004304  016200  025540                  MOV     AVAIL(R2),R0    ;CONTROL BLOCK ADDR IN R0
2072  004310  105760  000022                  TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR THE DRIVE ?
2073  004314  001403                          BEQ     2$              ;BR IF NOT
2074  004316  004737  006322                  JSR     PC,WRTPK        ;GET DATA PACK PARAMETERS
2075  004322  000402                          BR      3$              ;GET THE BUFFER
2076  004324  004737  005502          2$:     JSR     PC,GETPAR       ;LOAD NEW PARAMETERS
2077  004330  005046          3$:     CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
2078  004332  004737  004710                  JSR     PC,GETBUF       ;GET BUFFER
2079  004336  012660  000006                  MOV     (SP)+,$BUF(R0)  ;MOVE BUFFER ADDR TO DPB
```

```
2080  004342  001755                        BEQ     1$                ;BR IF '0' ADDR (NO BUFFER)
2081  004344  004737  005274                JSR     PC,FILBUF         ;FILL THE BUFFER
2082  004350  005060  000064                CLR     $FAIR(R0)         ;CLEAR THE 'FAIRNESS' COUNT
2083  004354  004737  005424                JSR     PC,GODRIV         ;PUT CURRENT DPB IN DRIVER
2084  004360  012705  025452                MOV     #ORDERQ,R5        ;ADDRESS OF ORDER QUEUE IN R5
2085  004364  005725                        TST     (R5)+             ;END OF QUEUE ?
2086  004366  001376                        BNE     .-2               ;BR IF NOT
2087  004370  010045                        MOV     R0,-(R5)          ;PUT BLOCK ADDRESS INTO QUEUE
2088  004372  000413                        BR      8$                ;COMPRESS THE 'AVAIL' LIST
2089  004374  005360  000064        5$:     DEC     $FAIR(R0)         ;DECREMENT THE 'FAIRNESS' COUNT
2090  004400  001402                        BEQ     6$                ;ENTRY WAITING FOR BUFFER PAST LIMIT
2091  004402  005722                        TST     (R2)+             ;INCREMENT THE 'AVAILABLE TABLE' POINTER
2092  004404  000734                        BR      1$                ;CONTINUE LOOKING
2093  004406  012705  025562        6$:     MOV     #WAIT,R5          ;TABLE ADDRESS
2094  004412  005725                        TST     (R5)+             ;AT END OF TABLE ?
2095  004414  001376                        BNE     .-2               ;BR IF NOT
2096  004416  016245  025540        7$:     MOV     AVAIL(R2),-(R5)   ;MOVE ENTRY FROM AVAILABLE TO WAIT
2097  004422  012701  025540        8$:     MOV     #AVAIL,R1         ;TABLE TO COMPRESS
2098  004426  060201                        ADD     R2,R1             ;FORM TABLE POSITION
2099  004430  004737  006640                JSR     PC,CMPRES         ;COMPRESS THE INDICATED TABLE
2100  004434  000720                        BR      1$                ;CONTINUE PROCESSING
2101
2102                                  ;GET BUFFER ASSIGNMENTS FOR DRIVES IN THE 'BUFFER WAIT' QUEUE
2103
2104  004436  013700  025562        MAIN3:  MOV     WAIT,R0           ;MOVE THE 'WAIT' ENTRY TO R0
2105  004442  005046                        CLR     -(SP)             ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
2106  004444  004737  004710                JSR     PC,GETBUF         ;TRY TO GET A BUFFER
2107  004450  012660  000006                MOV     (SP)+,$BUF(R0)    ;MOVE THE BUFFER ADDR TO THE DPB
2108  004454  001002                        BNE     1$                ;BR IF A BUFFER WAS ASSIGNED
2109  004456  000137  004522                JMP     IDLE              ;NO BUFFER AVAILABLE YET
2110  004462  004737  005274        1$:     JSR     PC,FILBUF         ;FILL THE BUFFER
2111  004466  004737  005424                JSR     PC,GODRIV         ;PUT THE ENTRY IN THE DRIVER
2112  004472  005060  000064                CLR     $FAIR(R0)         ;CLEAR THE 'FAIRNESS' COUNT
2113  004476  012705  025452                MOV     #ORDERQ,R5        ;ADDRESS OF ORDER QUEUE IN R5
2114  004502  005725                        TST     (R5)+             ;AT END OF THE QUEUE
2115  004504  001376                        BNE     .-2               ;BR IF NOT
2116  004506  010045                        MOV     R0,-(R5)          ;PUT BLOCK ADDRESS IN QUEUE
2117  004510  012701  025562                MOV     #WAIT,R1          ;TABLE ADDRESS
2118  004514  004737  006640                JSR     PC,CMPRES         ;COMPRESS THE WAIT TABLE
2119  004520  000662                        BR      MAIN2             ;LOOK FOR MORE ENTRIES
2120
2121                                  ;WAIT FOR AN ORDER TO FINISH
2122
2123  004522  012701  025452        IDLE:   MOV     #ORDERQ,R1        ;ADDRESS OF THE ORDER QUEUE IN R1
2124  004526  012100                1$:     MOV     (R1)+,R0          ;PUT BLOCK ADDRESS INTO R0
2125  004530  001430                        BEQ     3$                ;BR IF END OF QUEUE
2126  004532  005760  000016                TST     $STATUS(R0)       ;SEE IF UNIT FINISHED
2127  004536  001773                        BEQ     1$                ;BR IF UNIT NOT FINISHED
2128  004540  162701  000002                SUB     #2,R1             ;CORRECT THE QUEUE POINTER
2129  004544  010146                        MOV     R1,-(SP)          ;SAVE THE QUEUE ADDRESS
2130  004546  004737  004616                JSR     PC,STATIS         ;ACCUMULATE STATISTICS FOR UNIT IN R0
2131  004552  004737  012364                JSR     PC,PROCES         ;PROCESS END OF ORDER
2132  004556  004737  005046                JSR     PC,RELBUF         ;RESTORE THE BUFFER
2133  004562  004737  011276                JSR     PC,ABNRML         ;SEE IF ANY UNITS HAVE TOO MANY ERRORS
2134  004566  004737  011324                JSR     PC,NRML           ;SEE IF ANY UNIT HAS XFERED 3X10^9 BITS
2135  004572  012601                        MOV     (SP)+,R1          ;RESTORE THE ORDER TABLE INDEX
```

```
2136  004574  012705  025540                  MOV     #AVAIL,R5       ;FIND THE END OF THE 'AVAILABLE' TABLE
2137  004600  005725                  2$:     TST     (R5)+           ;END OF THE TABLE ?
2138  004602  001376                          BNE     2$              ;BR IF NOT AT END OF LIST
2139  004604  011145                          MOV     (R1), -(R5)     ;MOV THE BLOCK ADDRESS INTO THE TABLE
2140  004606  004737  006640                  JSR     PC,CMPRES       ;COMPRESS THE ORDER QUEUE
2141  004612  000137  003650          3$:     JMP     MAIN            ;CONTINUE THE LOOP
2142
2143                                  ;ROUTINE TO UPDATE THE GENERAL STATISTICS FOR THE DRIVE IN R0
2144
2145  004616  016037  000076  004706 STATIS: MOV     $RPBA(R0),FACTOR  ;STORE THE FINAL BUFFER ADDRESS
2146  004624  166037  000006  004706         SUB     $BUF(R0),FACTOR ;SUBTRACT THE INITIAL ADDRESS
2147  004632  001413                          BEQ     1$              ;BR IF NO DATA TRANSFER
2148  004634  006237  004706                  ASR     FACTOR          ;CONVERT TO A WORD COUNT
2149  004640  132760  000004  000002         BITB    #BIT02,$COMND(R0)  ;READ OR WRITE ORDER ?
2150  004646  001405                          BEQ     1$              ;BR IF WRITE
2151  004650  063760  004706  000044         ADD     FACTOR,$READ(R0)  ;UPDATE THE READ WORD COUNT
2152  004656  005560  000046                  ADC     $READ+2(R0)
2153  004662  026060  000010  000106 1$:     CMP     $CYL(R0),$SUCA(R0)  ;DID MID-TRANSFER SEEK OCCUR
2154  004670  001405                          BEQ     2$              ;BR IF NOT
2155  004672  062760  000001  000040         ADD     #1,$POSIT(R0)   ;INCREMENT SEEK COUNT
2156  004700  005560  000042                  ADC     $POSIT+2(R0)    ;ADD CARRY TO UPPER WORD
2157  004704  000207                  2$:     RTS     PC
2158
2159  004706  000000                  FACTOR: .WORD   0               ;USED FOR WORDS TRANSFERED
2160
2161                                  ;;*****************************************************************
2162
2163                                  .SBTTL  BUFFER SUPPORT ROUTINES
2164
2165                                  ;;*****************************************************************
2166
2167                                  ;ROUTINE TO GET A BUFFER
2168
2169  004710  010146                  GETBUF: MOV     R1,-(SP)        ;SAVE R1
2170  004712  010246                          MOV     R2,-(SP)        ;SAVE R2
2171  004714  010346                          MOV     R3,-(SP)        ;SAVE R3
2172  004716  013702  025604                  MOV     BUFTBL,R2       ;NUMBER OF SEPARATE BUFFERS
2173  004722  001413                          BEQ     5$              ;BR IF NONE AVAILABLE
2174  004724  012701  025606                  MOV     #BUFTBL+2,R1    ;FIRST ADDRESS OF ALLOCATION TABLE
2175  004730  026061  000020  000002 1$:     CMP     $WRDL(R0),2(R1) ;SEE IF THERE IS A BLOCK LARGE ENOUGH
2176  004736  101411                          BLOS    3$              ;BRANCH IF IT IS
2177  004740  005302                          DEC     R2              ;DECREMENT TABLE COUNT
2178  004742  001403                          BEQ     5$              ;BR IF THROUGH TABLE
2179  004744  062701  000004                  ADD     #4,R1           ;INCREMENT TABLE POINTER
2180  004750  000767                          BR      1$              ;CONTINUE LOOKING
2181  004752  012603                  5$:     MOV     (SP)+,R3        ;RESTORE R3
2182  004754  012602                          MOV     (SP)+,R2        ;RESTORE R2
2183  004756  012601                          MOV     (SP)+,R1        ;RESTORE R1
2184  004760  000207                          RTS     PC              ;RETURN
2185  004762  011166  000010          3$:     MOV     (R1),10(SP)     ;BUFFER ADDRESS TO STACK
2186  004766  166061  000020  000002         SUB     $WRDL(R0),2(R1) ;ADJUST BUFFER SIZE
2187  004774  001407                          BEQ     4$              ;BR IF DIFFERENCE IS ZERO
2188  004776  006360  000020                  ASL     $WRDL(R0)       ;CONVERT # WORDS TO BYTES
2189  005002  066011  000020                  ADD     $WRDL(R0),(R1)  ;MAKE NEW STARTING ADDRESS
2190  005006  006260  000020                  ASR     $WRDL(R0)       ;RETURN # BYTES TO WORDS
2191  005012  000757                          BR      5$              ;RETURN
```

```
 2192  005014  005337  025604          4$:     DEC     BUFTBL          ;DECREMENT ENTRIES COUNT
 2193  005020  001754                          BEQ     5$              ;BR IF ALLOCATION TABLE EMPTY
 2194  005022  005302                          DEC     R2              ;DECREMENT TABLE COUNT
 2195  005024  001752                          BEQ     5$              ;BR IF ITEM WERE LAST ENTRY
 2196  005026  010103                          MOV     R1,R3           ;MOVE TABLE POINTER
 2197  005030  062703  000004                  ADD     #4,R3           ;POINT TO NEXT ENTRY
 2198  005034  012321                  6$:     MOV     (R3)+,(R1)+     ;MOVE ITEM
 2199  005036  012321                          MOV     (R3)+,(R1)+
 2200  005040  005302                          DEC     R2              ;DECREMENT TABLE COUNT
 2201  005042  001374                          BNE     6$              ;CONTINUE IF NOT AT END OF TABLE
 2202  005044  000742                          BR      5$              ;RETURN
 2203
```

```
2204
2205                                    ;ROUTINE TO PUT BUFFER BACK IN TABLE
2206
2207  005046  104412                  RELBUF: SAVREG                          ;SAVE R0-R5
2208  005050  012701  025606                  MOV     #BUFTBL+2,R1            ;BEGINNING OF TABLE
2209  005054  013702  025604                  MOV     BUFTBL,R2               ;ENTRY COUNT
2210  005060  001424                          BEQ     2$                      ;BR IF EMPTY TABLE
2211  005062  016003  000020                  MOV     $WRDL(R0),R3            ;TRIAL ADDRESS
2212  005066  006303                          ASL     R3                      ;CHANGE TO BYTE COUNT
2213  005070  066003  000006                  ADD     $BUF(R0),R3             ;ADDRESS OF HIGHER ADJACENT BLOCK
2214  005074  021103                  1$:     CMP     (R1),R3                 ;UPPER ADJACENT BLOCK
2215  005076  001432                          BEQ     4$                      ;BR IF YES
2216  005100  062701  000004                  ADD     #4,R1                   ;INCREMENT POINTER
2217  005104  005302                          DEC     R2                      ;DECREMENT ENTRY COUNT
2218  005106  001372                          BNE     1$                      ;CONTINUE SEARCHING
2219  005110  016011  000006                  MOV     $BUF(R0),(R1)           ;PUT THE BUFFER BLOCK INTO THE TABLE
2220  005114  016061  000020  000002          MOV     $WRDL(R0),2(R1)          ;BLOCK SIZE
2221  005122  005237  025604                  INC     BUFTBL                  ;INCREMENT ENTRY COUNT
2222  005126  005202                          INC     R2                      ;INCREMENT R2 FOR USE LATER
2223  005130  000422                          BR      5$                      ;SEE IF A LOWER ADJACENT BLOCK IS IN THE TABLE
2224  005132  016021  000006          2$:     MOV     $BUF(R0),(R1)+          ;BLOCK ADDRESS TO TABLE
2225  005136  016021  000020                  MOV     $WRDL(R0),(R1)+         ;SIZE TO TABLE
2226
2227  005142  005237  025604                  INC     BUFTBL                  ;INCREMENT ENTRY COUNT
2228  005146  022737  000002  025604  3$:     CMP     #2,BUFTBL               ;SEE HOW MANY BLOCKS ARE IN TABLE
2229  005154  103001                          BHIS    .+4                     ;BR IF NOT MORE THAN 2
2230  005156  000240                          NOP                             ;DEBUGGING AID
2231
2232
2233  005160  104413                          RESREG                          ;RESTORE R0-R5
2234  005162  000207                          RTS     PC                      ;RETURN
2235  005164  016011  000006          4$:     MOV     $BUF(R0),(R1)           ;RELEASED BUFFER IS LOWER ADJACENT
2236  005170  066061  000020  000002          ADD     $WRDL(R0),2(R1)         ;INCREMENTED SIZE
2237  005176  010246                  5$:     MOV     R2,-(SP)                ;SAVE R2
2238  005200  013702  025604                  MOV     BUFTBL,R2               ;ENTRY COUNT
2239  005204  012705  025606                  MOV     #BUFTBL+2,R5            ;BEGINNING OF TABLE
2240  005210  016504  000002          6$:     MOV     2(R5),R4                ;BLOCK SIZE (IN WORDS)
2241  005214  006304                          ASL     R4                      ;CHANGE TO BYTE COUNT
2242  005216  061504                          ADD     (R5),R4                 ;ADD BLOCK BEGINNING ADDRESS
2243  005220  020411                          CMP     R4,(R1)                 ;R1 STILL POINTS TO INSERTED ENTRY
2244  005222  001406                          BEQ     8$                      ;LOWER ADJACENT IN TABLE
2245  005224  062705  000004                  ADD     #4,R5                   ;INCREMENT POINTER
2246  005230  005302                          DEC     R2                      ;DECREMENT ENTRY COUNT
2247  005232  001366                          BNE     6$                      ;CONTINUE LOOKING
2248  005234  005726                          TST     (SP)+                   ;RESTORE STACK POINTER
2249  005236  000743                          BR      3$                      ;END
2250  005240  012602                  8$:     MOV     (SP)+,R2                ;RESTORE R2
2251  005242  066165  000002  000002          ADD     2(R1),2(R5)             ;INCREMENT LOWER BLOCK LENGTH
2252  005250  005337  025604                  DEC     BUFTBL                  ;DECREMENT ENTRY COUNT
2253  005254  010105                          MOV     R1,R5                   ;GET READY TO COMPRESS
2254  005256  062705  000004                  ADD     #4,R5                   ;INCREMENT TO NEXT ENTRY
2255  005262  012521                  9$:     MOV     (R5)+,(R1)+             ;COMPRESS TABLE
2256  005264  012521                          MOV     (R5)+,(R1)+             ;MOVE SIZE FIELD DOWN
2257  005266  005302                          DEC     R2                      ;DECREMENT ENTRY COUNT
2258  005270  001374                          BNE     9$                      ;BR IF NOT FINISHED
2259  005272  000725                          BR      3$                      ;FINISHED
```

```
2260
2261
2262                                    ;FILL THE ASSIGNED BUFFER (IF WRITE OR WRITE CHECK ORDER)
2263
2264  005274  104412                    FILBUF: SAVREG                          ;SAVE R0-R5
2265  005276  132760  000004  000002            BITB    #BIT02,$COMND(R0)       ;READ OR WRITE ORDER ?
2266  005304  001404                            BEQ     1$                      ;BR IF WRITE
2267  005306  122760  000007  000002            CMPB    #WRTCHK,$COMND(R0)      ;WRITE CHECK ?
2268  005314  001041                            BNE     3$                      ;BR IF NOT
2269  005316  026027  000006  033634  1$:       CMP     $BUF(R0),#ENDPGM        ;MAKE SURE BUFFER IS ABOVE PROGRAM
2270  005324  103002                            BHIS    .+6                     ;BR IF IT IS
2271  005326  000000                            HALT                            ;BUFFER ADDRESS WITHIN PROGRAM, ILLEGAL
2272  005330  000776                            BR      .-2                     ;INTERLOCK THE HALT
2273  005332  023760  001306  000020            CMP     BUFRSZ,$WRDL(R0)        ;SEE IF WORD LENGTH DOESN'T EXCEED MAXIMUM
2274  005340  103002                            BHIS    .+6                     ;BR IF IT DOESN'T
2275  005342  000000                            HALT                            ;WORD LENGTH FOR OPERATION GREATER THAN AVAIL MEM
2276  005344  000776                            BR      .-2                     ;INTERLOCK THE HALT
2277  005346  016001  000006                    MOV     $BUF(R0),R1             ;BUFFER ADDRESS
2278  005352  016002  000020                    MOV     $WRDL(R0),R2            ;POSITIVE WORD COUNT
2279  005356  005004                            CLR     R4                      ;CLEAR R4
2280  005360  116004  000026                    MOVB    $PATTC(R0),R4           ;RELATIVE PATTERN ADDRESS
2281  005364  016405  025754                    MOV     STNDAT(R4),R5           ;PATTERN ADDRESS
2282  005370  012703  000020                    MOV     #16.,R3                 ;PATTERN COUNT
2283  005374  012521                    2$:     MOV     (R5)+,(R1)+             ;MOVE THE PATTERN INTO THE BUFFER
2284  005376  005302                            DEC     R2                      ;DECREMENT THE WORD COUNT
2285  005400  001407                            BEQ     3$                      ;BR IF DONE (WORD COUNT = 0)
2286  005402  005303                            DEC     R3                      ;DECREMENT THE PATTERN COUNT
2287  005404  001373                            BNE     2$                      ;BR IF MORE PATTERN
2288  005406  012703  000020                    MOV     #16.,R3                 ;RESTORE PATTERN COUNT
2289  005412  016405  025754                    MOV     STNDAT(R4),R5           ;RESTORE THE ADDRESS
2290  005416  000766                            BR      2$                      ;CONTINUE DISTRIBUTING THE PATTERN
2291  005420  104413                    3$:     RESREG                          ;RESTORE R0-R5
2292  005422  000207                            RTS     PC                      ;RETURN
2293
2294                                    ;START THE ORDER FOR THE DPB IN R0
2295
2296  005424  010046                    GODRIV: MOV     R0,-(SP)                ;SAVE R0
2297  005426  010037  005436                    MOV     R0,1$                   ;CURRENT DPB ADDRESS
2298  005432  004037  021556                    JSR     R0,RP11                 ;DRIVER ENTRANCE
2299  005436  000000                    1$:     0                               ;DPB ADDR GOES HERE
2300  005440  000000                            HALT                            ;DRIVER REJECTED REQUEST
2301  005442  012600                            MOV     (SP)+,R0                ;RESTORE R0
2302  005444  062760  000001  000034            ADD     #1,$OPERC(R0)           ;INCREMENT THE OPERATION COUNT
2303  005452  005560  000036                    ADC     $OPERC+2(R0)            ;ADD ANY CARRY
2304  005456  026060  000032  000010            CMP     $PREVA+2(R0),$CYL(R0)   ;DID ORDER REQUIRE A CYLINDER CHANGE
2305  005464  001405                            BEQ     2$                      ;BR IF NOT
2306  005466  062760  000001  000040            ADD     #1,$POSIT(R0)           ;INCREMENT SEEK COUNT
2307  005474  005560  000042                    ADC     $POSIT+2(R0)            ;ADD ANY CARRY
2308  005500  000207                    2$:     RTS     PC
2309
2310                                    ;;*****************************************************************
2311
2312                                    .SBTTL  GENERATE VARIABLES FOR THE NEXT OPERATION
2313
2314                                    ;;*************** ***********************************************
2315
```

GENERATE VARIABLES FOR THE NEXT OPERATION SEQ 0044

```
2316                                    ;GENERATE VARIABLES FOR THE OPERATION
2317
2318  005502  010546                    GETPAR: MOV     R5,-(SP)          ;SAVE R5
2319  005504  004737  006606                    JSR     PC,PREVST         ;SAVE CURRENT SECTOR & TRACK ADDRS
2320  005510  016060  000106  000032            MOV     $SUCA(R0),$PREVA+2(R0)  ;SAVE CURRENT CYLINDER
2321  005516  032777  000040  173414            BIT     #SW05,@SWR        ;SWITCH 5 SET ?
2322  005524  001414                            BEQ     GTPAR1            ;BR IF NOT SET
2323  005526  122760  000007  000002            CMPB    #WRTCHK,$COMND(R0)  ;PRESENT ORDER A WRITE CHECK ?
2324  005534  001006                            BNE     1$                ;BR IF NOT
2325  005536  005737  001322                    TST     AUTOCK            ;AUTO WRITE CHECKS ?
2326  005542  001403                            BEQ     1$                ;BR IF NOT
2327  005544  116060  000024  000002            MOVB    $PREVO(R0),$COMND(R0)  ;RESTORE PREVIOUS OPERATION
2328  005552  000137  006206            1$:     JMP     PARXIT            ;EXIT
2329  005556  116060  000002  000024  GTPAR1: MOVB    $COMND(R0),$PREVO(R0)  ;SAVE CURRENT OPERATION CODE
2330  005564  004737  020512                    JSR     PC,$RAND          ;CYCLE THE RANDOM NUMBER GENERATOR
2331  005570  032777  000001  173342            BIT     #SW0,@SWR         ;SEE IF SW0 SET
2332  005576  001017                            BNE     2$                ;BR IF SET - READ ONLY
2333  005600  012705  000010            1$:     MOV     #10,R5            ;READ/WRITE SELECTION DIVISOR
2334  005604  004737  011570                    JSR     PC,GETREM         ;GET SELECTION VALUE
2335  005610  020537  001320                    CMP     R5,RATIO          ;DETERMINE IF READ OR WRITE
2336  005614  003010                            BGT     2$                ;BR IF READ
2337  005616  004737  006220                    JSR     PC,RANWRT         ;SELECT A WRITE ORDER
2338  005622  122760  000007  000002            CMPB    #WRTCHK,$COMND(R0)  ;WAS WRITE CHECK SELECTED ?
2339  005630  001013                            BNE     GETSEC            ;NO, CONTINUE WITH THE SELECTION
2340  005632  000137  006206                    JMP     PARXIT            ;BYPASS REST OF VARIABLE SELECTION
2341  005636  013705  020610            2$:     MOV     $HINUM,R5         ;SELECT READ OPERATION CODE
2342  005642  042705  177776                    BIC     #^C1,R5           ;MASK OUT ALL BUT BIT 0
2343  005646  062705  000003                    ADD     #3,R5             ;TABLE OFFSET FOR READ CODE
2344  005652  116560  025746  000002  3$:     MOVB    COMTBL(R5),$COMND(R0)  ;ORDER SELECTION CODE TO CONTROL BLOCK
2345
2346                                    ;GENERATE A RANDOM SECTOR ADDRESS BETWEEN VALUES 'FS' & 'LS'
2347
2348  005660  013705  001334            GETSEC: MOV     LS,R5   ;GET MAXIMUM SECTOR ADDRESS
2349  005664  163705  001336                    SUB     FS,R5   ;SUBTRACT MINIMUM SECTOR ADDRESS
2350  005670  001403                            BEQ     1$              ;BR IF MIN & MAX THE SAME
2351  005672  005205                            INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
2352  005674  004737  011570                    JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
2353  005700  063705  001336            1$:     ADD     FS,R5   ;CONVERT TO ADDRESS
2354  005704  110560  000012                    MOVB    R5,$SEC(R0)     ;STORE SECTOR ADDRESS IN DPB
2355
2356                                    ;GENERATE A RANDOM TRACK ADDRESS BETWEEN VALUES 'FT' & 'LT'
2357
2358  005710  013705  001330            GETRK:  MOV     LT,R5   ;GET MAXIMUM TRACK ADDRESS
2359  005714  163705  001332                    SUB     FT,R5   ;SUBTRACT MINIMUM TRACK ADDRESS
2360  005720  001403                            BEQ     1$              ;BR IF MIN & MAX THE SAME
2361  005722  005205                            INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
2362  005724  004737  011570                    JSR     PC,GETREM       ;GET THE RANDOM TRACK AUGMENT
2363  005730  063705  001332            1$:     ADD     FT,R5   ;CONVERT AUGMENT TO AN ADDRESS
2364  005734  110560  000013                    MOVB    R5,$TRK(R0)     ;STORE TRACK ADDRESS IN DPB
2365                                            .
2366                                    ;GENERATE A RANDOM CYLINDER ADDRESS BETWEEN VALUES 'FC' & 'LC'
2367
2368  005740  013705  001324            GETCYL: MOV     LC,R5   ;GET MAXIMUM CYLINDER ADDRESS
2369  005744  163705  001326                    SUB     FC,R5   ;SUBTRACT MINIMUM CYLINDER ADDRESS
2370  005750  001403                            BEQ     1$              ;BR IF MIN & MAX THE SAME
2371  005752  005205                            INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
```

```
                        GENERATE VARIABLES FOR THE NEXT OPERATION                                        SEQ 0045

 2372  005754  004737  011570                 JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
 2373  005760  063705  001326         1$:     ADD     R0,R5   ;CONVERT AUGMENT TO AN ADDRESS
 2374  005764  010560  000010                 MOV     R5,$CYL(R0)     ;;STORE CYLINDER ADDRESS IN DPB
 2375
 2376                                 ;GENERATE A RANDOM BUFFER LENGTH BETWEEN 4 & THE VALUE IN 'BUFRSZ'
 2377
 2378  005770  013705  001306         GETSIZ: MOV     BUFRSZ,R5       ;GET BUFFER SIZE
 2379  005774  005205                 INC     R5              ;INCREMENT THE MAXIMUM SIZE
 2380  005776  004737  011570                 JSR     PC,GETREM       ;DIVIDE BY MAX VALUE
 2381  006002  005705                 TST     R5              ;IS THE REMAINDER 0 ?
 2382  006004  001413                 BEQ     1$              ;ZERO, TRY AGAIN
 2383  006006  010546                 MOV     R5,-(SP)        ;NEW WORD LENGTH ON STACK FOR CHECK
 2384  006010  005046                 CLR     -(SP)           ;MAKE UPPER DIVIDEND ZERO
 2385  006012  012746  000400                 MOV     #256.,-(SP)     ;SECTOR SIZE IS THE DIVISOR
 2386  006016  004737  020176                 JSR     PC,$DIV         ;DIVIDE BUFFER SIZE BY SECTOR SIZE
 2387  006022  021627  000004                 CMP     (SP),#4         ;SEE IF REMAINDER LESS THAN 4
 2388  006026  103005                 BHIS    2$              ;BR IF NOT
 2389  006030  062706  000004                 ADD     #4,SP           ;CORRECT THE STACK POINTER
 2390  006034  004737  020512         1$:     JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
 2391  006040  000753                 BR      GETSIZ          ;TRY AGAIN
 2392  006042  132760  000004  000002 2$:     BITB    #4,$COMND(R0)   ;READ OR WRITE ?
 2393  006050  001002                 BNE     3$              ;BR IF READ
 2394  006052  042705  000001                 BIC     #1,R5           ;MAKE THE SIZE EVEN
 2395  006056  010560  000020         3$:     MOV     R5,$WRDL(R0)    ;WORD LENGTH TO CONTROL BLOCK
 2396  006062  122760  000013  000002         CMPB    #WRITE,$COMND(R0)  ;WRITE WITHOUT IMPLIED SEEK ?
 2397  006070  001404                 BEQ     4$              ;BR IF IT IS
 2398  006072  122760  000017  000002         CMPB    #READ,$COMND(R0)  ;READ WITHOUT IMPLIED SEEK ?
 2399  006100  001027                 BNE     6$              ;BR IF NOT
 2400  006102  005016         4$:     CLR     (SP)            ;CLEAR THE STACK
 2401  006104  116016  000012                 MOVB    $SEC(R0),(SP)   ;MOVE THE SECTOR ADDRESS
 2402  006110  066616  000002                 ADD     2(SP),(SP)      ;ADD THE NUMBER OF SECTORS IN THE BUFFER
 2403  006114  022716  000010                 CMP     #8.,(SP)        ;DO THEY EXCEED THE TRACK
 2404  006120  103017                 BHIS    6$              ;BR IF NOT
 2405  006122  132760  000010  000002         BITB    #10,$COMND(R0)  ;SEE IF AN 'IMPLIED SEEK' COMMAND
 2406  006130  001413                 BEQ     6$              ;BR IF NOT
 2407  006132  132760  000004  000002         BITB    #4,$COMND(R0)   ;READ WITHOUT IMPLIED SEEK ?
 2408  006140  001004                 BNE     5$              ;BR IF READ
 2409  006142  112760  000003  000002         MOVB    #WRTSEK,$COMND(R0)  ;LOAD COMMAND
 2410  006150  000403                 BR      6$              ;CONTINUE
 2411  006152  112760  000005  000002 5$:     MOVB    #RDSEK,$COMND(R0)  ;LOAD COMMAND
 2412  006160  062706  000004         6$:     ADD     #4,SP           ;CORRECT THE STACK POINTER
 2413  006164  016060  000020  000004         MOV     $WRDL(R0),$WRDM(R0)  ;SETUP WORD COUNT
 2414  006172  005460  000004                 NEG     $WRDM(R0)       ;MAKE IT NEGATIVE
 2415
 2416                                 ;GET THE PATTERN CODE & EXIT FROM PARAMETER SELECTION
 2417
 2418  006176  004737  006274         PATCOD: JSR     PC,GETPAT       ;GET PATTERN CODE
 2419  006202  110560  000026                 MOVB    R5,$PATTC(R0)   ;MOVE PATTERN CODE TO CONTROL BLOCK
 2420  006206  013760  001254  000064 PARXIT: MOV     FAIRNS,$FAIR(R0)  ;LOAD 'FAIRNESS' COUNT
 2421  006214  012605                 MOV     (SP)+,R5        ;RESTORE R5
 2422  006216  000207                 RTS     PC              ;RETURN
 2423
 2424                                 ;ROUTINE TO SELECT A WRITE (OR WRITE CHECK) OPERATION
 2425
 2426  006220  012705  000003         RANWRT: MOV     #3,R5           ;WRITE OPERATION SELECTION DIVISOR
 2427  006224  004737  011570                 JSR     PC,GETREM       ;GET SELECTION CODE
```

```
 2428  006230  005737  001322                    TST     AUTOCK          ;ARE WRITE CHECK ORDERS TO BE SELECTED
 2429                                                                      ;RANDOMLY ?
 2430  006234  001403                            BEQ     1$              ;BR IF THEY ARE
 2431  006236  042705  000002                    BIC     #2,R5           ;SET CODE TO EXCLUDE WRITE CHECK ORDERS
 2432  006242  000410                            BR      2$              ;FINISH
 2433  006244  020527  000002          1$:       CMP     R5,#2           ;WRITE CHECK SELECTED ?
 2434  006250  001005                            BNE     2$              ;BR IF NOT
 2435  006252  132760  000004  000024            BITB    #BIT02,$PREVO(R0)  ;WAS PREVIOUS ORDER A WRITE ?
 2436  006260  001401                            BEQ     2$              ;BR IF IT WAS
 2437  006262  005005                            CLR     R5              ;CHANGE WRITE CHECK CODE TO WRITE WITH
 2438                                                                      ;IMPLIED SEEK
 2439  006264  116560  025746  000002  2$:       MOVB    COMTBL(R5),$COMND(R0)  ;CODE TO DRIVE BLOCK
 2440  006272  000207                            RTS     PC              ;RETURN
 2441
 2442                                  ;ROUTINE TO SELECT A PATTERN
 2443
 2444  006274  012705  000020          GETPAT: MOV     #16.,R5         ;SELECT PATTERN
 2445  006300  004737  011570                    JSR     PC,GETREM       ;GET CODE
 2446  006304  005705                            TST     R5              ;WAS PATTERN ZERO SELECTED ?
 2447  006306  001003                            BNE     1$              ;BR IF NOT ZERO
 2448  006310  004737  020512                    JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
 2449  006314  000767                            BR      GETPAT          ;TRY AGAIN
 2450  006316  006305          1$:       ASL     R5              ;MAKE CODE INTO TABLE INDEX
 2451  006320  000207                            RTS     PC              ;RETURN
 2452
 2453                                  ;ROUTINE TO SETUP PARAMETERS FOR A SEQUENTIAL READ OR WRITE OF THE DISK
 2454
 2455  006322  004737  020512          WRTPK:  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
 2456  006326  005760  000036                    TST     $OPERC+2(R0)    ;SEE IF FIRST OPERATION
 2457  006332  001007                            BNE     WRTPK1          ;BR IF UPPER WORD OF COUNTER NOT ZERO
 2458  006334  005760  000034                    TST     $OPERC(R0)      ;LOWER WORD ZERO ?
 2459  006340  001004                            BNE     WRTPK1          ;BR IF NOT 1ST OPERATION
 2460  006342  105760  000022                    TSTB    $PACK(R0)       ;SEE WHICH - 'R' OR 'W'
 2461  006346  100504                            BMI     WRTPK3          ;BR IF 'W'
 2462  006350  000477                            BR      WRTPK2          ;'R' OPERATION
 2463  006352  032777  000040  172560  WRTPK1: BIT     #SW05,@SWR      ;LOOP ON LAST ADDRESS ?
 2464  006360  001106                            BNE     WRTPK4          ;BR IF LOOP
 2465  006362  032760  000002  000070            BIT     #EOP,$RPER(R0)  ;DID LAST ORDER OVERFLOW ?
 2466  006370  001032                            BNE     1$              ;BR IF IT DID
 2467  006372  116060  000002  000024            MOVB    $COMND(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
 2468  006400  004737  006606                    JSR     PC,PREVST       ;SAVE CURRENT SECTOR & TRACK
 2469  006404  016060  000106  000032            MOV     $SUCA(R0),$PREVA+2(R0)  ;CURRENT CYLINDER
 2470  006412  016060  000102  000012            MOV     $RPDA(R0),$SEC(R0)  ;NEW SECTOR & TRACK ADDRESS
 2471  006420  042760  000360  000012            BIC     #360,$SEC(R0)   ;CLEAR THE 'SOT' BITS
 2472  006426  016060  000100  000010            MOV     $RPCA(R0),$CYL(R0)  ;NEW CYLINDER ADDRESS
 2473  006434  026037  000010  001324            CMP     $CYL(R0),LC     ;SEE IF END CYLINDER
 2474  006442  103427                            BLO     2$              ;BR IF NOT
 2475  006444  101004                            BHI     1$              ;BR IF PAST END CYLINDER
 2476  006446  126037  000013  001330            CMPB    $TRK(R0),LT     ;END TRACK ?
 2477  006454  101422                            BLOS    2$              ;BR IF NOT EXCEEDED
 2478  006456  113760  001332  000013  1$:       MOVB    FT,$TRK(R0)     ;RESET TRACK ADDRESS
 2479  006464  113760  001336  000012            MOVB    FS,$SEC(R0)  ;RESET SECTOR ADDRESS
 2480  006472  013760  001326  000010            MOV     FC,$CYL(R0)  ;RESET CYLINDER ADDRESS
 2481  006500  004737  011400                    JSR     PC,NRML2        ;DROP THE UNIT (NORMAL TERMINATION)
 2482  006504  032777  000020  172426            BIT     #20,@SWR        ;IS SWITCH 4 SET ?
 2483  006512  001003                            BNE     2$              ;BR IF SET
```

GENERATE VARIABLES FOR THE NEXT OPERATION SEQ 0047

```
2484  006514  005726                          TST     (SP)+           ;INCREMENT THE STACK POINTER
2485  006516  000137  003650                  JMP     MAIN            ;RETURN DIRECTLY TO 'MAIN'
2486  006522  013760  001306  000020  2$:     MOV     BUFRSZ,$WRDL(R0)  ;BUFFER SIZE IS MAXIMUM
2487  006530  013760  001306  000004          MOV     BUFRSZ,$WRDM(R0)  ;WORD COUNT
2488  006536  005460  000004                  NEG     $WRDM(R0)       ;CHANGE WORD COUNT TO 2'S COMPLEMENT
2489  006542  105760  000022                  TSTB    $PACK(R0)       ;READ OR WRITE ?
2490  006546  100404                          BMI     WRTPK3          ;BR IF WRITE
2491  006550  112760  000005  000002  WRTPK2: MOVB    #RDSEK,$COMND(R0)  ;OP CODE
2492  006556  000407                          BR      WRTPK4          ;EXIT
2493  006560  112760  000003  000002  WRTPK3: MOVB    #WRTSEK,$COMND(R0)  ;OP CODE
2494  006566  004737  006274                  JSR     PC,GETPAT       ;GET PATTERN CODE
2495  006572  110560  000026                  MOVB    R5,$PATTC(R0)   ;PATTERN CODE
2496  006576  013760  001254  000064  WRTPK4: MOV     FAIRNS,$FAIR(R0)  ;LOAD 'FAIRNES' COUNT
2497  006604  000207                          RTS     PC              ;RETURN
2498
2499                                  ;DECREMENT AND SAVE CURRENT SECTOR & TRACK ADDRESS
2500
2501  006606  162706  000004          PREVST: SUB     #4,SP           ;MAKE ROOM ON THE STACK
2502  006612  004737  006656                  JSR     PC,READDR       ;DECREMENT SECTOR-TRACK ADDRESS
2503  006616  112660  000031                  MOVB    (SP)+,$PREVA+1(R0)  ;SAVE CURRENT TRACK
2504  006622  112660  000030                  MOVB    (SP)+,$PREVA(R0)  ;SAVE CURRENT SECTOR
2505  006626  000207                          RTS     PC              ;RETURN
2506
2507                                  ;;*****************************************************************
2508
2509                                  .SBTTL  GENERAL SUPPORT SUBROUTINES
2510
2511                                  ;;*****************************************************************
2512
2513                                  ;DUMMY HANDLER IF 'CONTROL C' ENTERED DURING NORMAL OPERATION
2514
2515  006630  012746  000003          DORTI:  MOV     #3,-(SP)        ;PUT THE 'CONTROL C' BACK ON THE STACK
2516  006634  000137  017070                  JMP     $TKSRV+52       ;RETURN TO THE ENTRY ROUTINE
2517
2518                                  ;ROUTINE TO COMPRESS THE TABLE WHOSE ADDRESS IS IN R1
2519
2520  006640  016111  000002          CMPRES: MOV     2(R1),(R1)      ;COMPRESS THE TABLE IN R1
2521  006644  001403                          BEQ     1$              ;BR WHEN ZERO FOUND
2522  006646  062701  000002                  ADD     #2,R1           ;INCREMENT R1
2523  006652  000772                          BR      CMPRES          ;CONTINUE COMPRESSING TABLE
2524  006654  000207                  1$:     RTS     PC              ;RETURN
2525
2526                                  ;DECREMENT THE SECTOR-TRACK ADDRESS
2527
2528  006656  005066  000004          READDR: CLR     4(SP)           ;CLEAR STACK FOR SECTOR
2529  006662  116066  000102  000004          MOVB    $RPDA(R0),4(SP)  ;INCREMENTED SECTOR ON STACK
2530  006670  042766  177760  000004          BIC     #^C17,4(SP)     ;CLEAR THE 'SOT' BITS
2531  006676  105366  000004                  DECB    4(SP)           ;DECREMENT THE SECTOR ADDRESS
2532  006702  100017                          BPL     1$              ;BR IF SECTOR GREATER THAN 0
2533  006704  012766  000011  000004          MOV     #9.,4(SP)       ;JAM SECTOR ADDRESS TO 9
2534  006712  005066  000002                  CLR     2(SP)           ;CLEAR STACK FOR TRACK
2535  006716  116066  000103  000002          MOVB    $RPDA+1(R0),2(SP)  ;TRACK ADDRESS
2536  006724  105366  000002                  DECB    2(SP)           ;DECREMENT TRACK ADDRESS
2537  006730  100007                          BPL     2$              ;BR IF IT DIDN'T GO NEG
2538  006732  012766  000023  000002          MOV     #19.,2(SP)      ;RESET TRACK TO 19(10)
2539  006740  000403                          BR      2$
```

```
2540  006742  116066  000103  000002  1$:     MOVB    $RPDA+1(R0),2(SP)  ;TRACK ADDRESS
2541  006750  000207                  2$:     RTS     PC                 ;RETURN
2542
2543                                    ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
2544
2545  006752  012737  177777  001236  CKCLK:  MOV     #-1,CLKFLG         ;CLEAR CLOCK AVAILABILITY FLAG
2546  006760  012737  007034  000004          MOV     #CKCLK1,@#ERRVEC   ;SET UP VECTOR FOR CLOCK CHECK
2547  006766  005037  000006                  CLR     @#ERRVEC+2           ;NEW PSW
2548  006772  005777  172226                  TST     @$LKCSR            ;CHECK FOR KW11-P
2549  006776  005037  001236                  CLR     CLKFLG             ;SET CLOCK AVAILABILITY FLAG
2550  007002  013701  001230                  MOV     $LPVEC,R1          ;KW11-P VECTOR ADDRESS
2551  007006  012721  007650                  MOV     #CLOCK,(R1)+       ;SET UP KW11-P VECTOR
2552  007012  012711  000300                  MOV     #300,(R1)          ;PSW - PRI 6
2553  007016  012777  177777  172202          MOV     #-1,@$LKCSB        ;LOAD COUNTER BUFFER WITH 1'S
2554  007024  012777  000135  172172          MOV     #135,@$LKCSR       ;SET CLOCK - CNT UP, 16MS, CONT INT
2555  007032  000425                  BR      CKCLK3
2556  007034  062706  000004          CKCLK1: ADD     #4,SP              ;RESTORE THE STACK POINTER
2557  007040  012737  007102  000004          MOV     #CKCLK2,@#ERRVEC   ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
2558  007046  005777  172160                  TST     @$LKS              ;LOOK FOR KW11-L
2559  007052  005037  001236                  CLR     CLKFLG             ;SET CLOCK FLAG
2560  007056  013701  001234                  MOV     $LLVEC,R1          ;KW11-L VECTOR ADDRESS
2561  007062  012721  007650                  MOV     #CLOCK,(R1)+       ;SET UP KW11-L VECTOR
2562  007066  012711  000300                  MOV     #300,(R1)          ;PSW - PRI 6
2563  007072  012777  000100  172132          MOV     #100,@$LKS         ;SET KW11-L INTERRUPT
2564  007100  000402                  BR      CKCLK3
2565  007102  062706  000004          CKCLK2: ADD     #4,SP              ;RESTORE THE STACK POINTER
2566  007106  012737  000006  000004  CKCLK3: MOV     #6,@#ERRVEC        ;RESTORE THE ERROR VECTOR
2567  007114  000207                  RTS     PC
2568
2569                                    ;ROUTINE TO DISPLAY STATISTICS FOR ALL DRIVES ASSIGNED
2570
2571  007116  104412          STATPR: SAVREG                             ;SAVE R0-R5
2572  007120  005004                  CLR     R4                         ;R4 CONTAINS THE UNIT POINTER
2573  007122  012703  000010                  MOV     #8.,R3             ;R3 CONTAINS THE UNIT COUNTER
2574  007126  012700  024264                  MOV     #BLK0,R0           ;ADDR OF FIRST COUNTER
2575  007132  136437  021274  025450  1$:     BITB    ATABIT(R4),ASNLST  ;SEE IF DRIVE ASSIGNED
2576  007140  001006                  BNE     2$                         ;BR IF IT IS
2577  007142  005204                  INC     R4                         ;INCREMENT THE UNIT POINTER
2578  007144  062700  000112                  ADD     #$SILO+2,R0        ;INCREMENT THE BLOCK ADDRESS
2579  007150  005303                  DEC     R3                         ;FINISHED ?
2580  007152  001367                  BNE     1$                         ;BR IF NOT
2581  007154  000423                  BR      6$                         ;YES, EXIT
2582  007156  004737  007260  2$:     JSR     PC,SHDTYP                  ;TYPE THE HEADING
2583  007162  136437  021274  025450  3$:     BITB    ATABIT(R4),ASNLST  ;IS THE UNIT ASSIGNED
2584  007170  001003                  BNE     4$                         ;BR IF ASSIGNED
2585  007172  062700  000112                  ADD     #$SILO+2,R0        ;ADD TABLE SIZE TO ADDRESS
2586  007176  000407                  BR      5$
2587  007200  010002          4$:     MOV     R0,R2                      ;PUT BLOCK ADDRESS IN R2
2588  007202  062702  000034                  ADD     #$CPERC,R2         ;FIRST COUNTER TO DISPLAY
2589  007206  004737  007276                  JSR     PC,SDETAL          ;TYPE THE STATISTICS
2590  007212  062700  000112                  ADD     #$SILO+2,R0        ;INCREMENT BLOCK ADDRESS
2591  007216  005204          5$:     INC     R4                         ;INCREMENT UNIT NUMBER
2592  007220  005303                  DEC     R3                         ;DECREMENT UNIT COUNTER
2593  007222  001357                  BNE     3$                         ;BR IF NOT FINISHED
2594  007224  104413          6$:     RESREG                             ;RESTORE R0-R5
2595  007226  000207                  RTS     PC
```

```
 2596
 2597                                   ;ROUTINE TO TYPE STATISTICS FOR THE DRIVE IN R0
 2598
 2599  007230  104412                   TYPEST: SAVREG                          ;SAVE R0-R5
 2600  007232  004737  007260                   JSR     PC,SHDTYP               ;TYPE THE HEADING
 2601  007236  005004                           CLR     R4                      ;CLEAR R4 FOR DRIVE NUMBER
 2602  007240  111004                           MOVB    (R0),R4                 ;DRIVE NUMBER
 2603  007242  010002                           MOV     R0,R2                   ;PUT BLOCK ADDRESS IN R2
 2604  007244  062702  000034                   ADD     #$OPERC,R2              ;ADDRESS OF '$OPERC' IN PRESENT CLOCK
 2605  007250  004737  007276                   JSR     PC,SDETAL               ;TYPE THE STATISTICS
 2606  007254  104413                           RESREG                          ;RESTORE R0-R5
 2607  007256  000207                           RTS     PC                      ;RETURN
 2608
 2609                                   ;TYPE THE HEADER FOR THE DRIVE PERFORMANCE SUMMARY TYPEOUT
 2610
 2611  007260  104401  001213           SHDTYP: TYPE    ,$CRLF                  ;CR-LF
 2612  007264  004737  007624                   JSR     PC,$TIME                ;TYPE THE TIME OF DAY
 2613  007270  104401  032636                   TYPE    ,STATHD                 ;HEADER
 2614  007274  000207                           RTS     PC                      ;RETURN
 2615
 2616                                   ;TYPE THE DRIVE PERFORMANCE DATA LINE
 2617
 2618  007276  010446                   SDETAL: MOV     R4,-(SP)                ;DRIVE NUMBER
 2619  007300  104403                           TYPOS                           ;TYPE IT
 2620  007302     001     000                   .BYTE   1,0
 2621  007304  104401  032467                   TYPE    ,SPACE3                 ;SPACES
 2622  007310  016046  000062                   MOV     $PASSC(R0),-(SP)  ;PUT THE PASS COUNT ON THE STACK
 2623  007314  004737  015564                   JSR     PC,$SB2D                ;CONVERT IT
 2624  007320  004737  011632                   JSR     PC,$RPZR5               ;TYPE IT
 2625  007324  104401  032470                   TYPE    ,SPACE2                 ;SPACES
 2626  007330  010246                           MOV     R2,-(SP)                ;PUT $OPERC ON THE STACK
 2627  007332  004737  020614                   JSR     PC,$DB2D                ;CONVERT IT
 2628  007336  004737  011612                   JSR     PC,$RPZR8               ;TYPE $OPERC
 2629  007342  104401  032470                   TYPE    ,SPACE2                 ;SPACES
 2630  007346  062702  000004                   ADD     #4,R2                   ;INCREMENT R2
 2631  007352  010246                           MOV     R2,-(SP)                ;PUT $POSIT ON THE STACK
 2632  007354  004737  020614                   JSR     PC,$DB2D                ;CONVERT IT
 2633  007360  004737  011612                   JSR     PC,$RPZR8               ;TYPE $POSIT
 2634  007364  104401  032470                   TYPE    ,SPACE2                 ;SPACES
 2635  007370  062702  000004                   ADD     #4,R2                   ;INCREMENT R2
 2636  007374  010246                           MOV     R2,-(SP)                ;PUT $READ ON THE STACK
 2637  007376  004737  020614                   JSR     PC,$DB2D                ;CONVERT $READ
 2638  007402  004737  011642                   JSR     PC,$RPZR0               ;TYPE IT
 2639  007406  104401  032471                   TYPE    ,SPACE1                 ;1 SPACE
 2640  007412  062702  000004                   ADD     #4,R2                   ;INCREMENT R2
 2641  007416  010146                   1$:     MOV     R1,-(SP)                ;SAVE R1
 2642  007420  012701  000004                   MOV     #4,R1                   ;USE R1 AS A COUNTER
 2643  007424  062702  000002                   ADD     #2,R2                   ;INCREMENT R2 AGAIN
 2644  007430  012246                   2$:     MOV     (R2)+,-(SP)             ;PUT '$SOFT', '$HARD', '$SKI', OR '$MISPO'
 2645                                                                          ;ON THE STACK
 2646  007432  004737  015564                   JSR     PC,$SB2D                ;CONVERT IT
 2647  007436  004737  011622                   JSR     PC,$RPZR4               ;TYPEOUT
 2648  007442  104401  032471                   TYPE    ,SPACE1                 ;1 SPACE
 2649  007446  005301                           DEC     R1                      ;DONE ?
 2650  007450  001367                           BNE     2$                      ;BR IF NOT
 2651  007452  012601                           MOV     (SP)+,R1                ;RESTORE R1
```

```
 2652  007454  016046  000050                 MOV     $TOTAL(R0),-(SP)  ;CALCULATE NUMBER OF OTHER ERRORS
 2653  007460  166016  000052                 SUB     $SOFT(R0),(SP)  ;SUBTRACT $SOFT FROM $TOTAL
 2654  007464  166016  000054                 SUB     $HARD(R0),(SP)  ;SUBTRACT $HARD FROM $TOTAL
 2655  007470  166016  000056                 SUB     $SKI(R0),(SP)   ;SUBTRACT $SKI FROM $TOTAL
 2656  007474  166016  000060                 SUB     $MISPO(R0),(SP) ;SUBTRACT $MISPO FROM $TOTAL
 2657  007500  004737  015564                 JSR     PC,$SB2D        ;CONVERT 'OTHER' COUNT
 2658  007504  004737  011622                 JSR     PC,$RPZR4       ;TYPE IT
 2659  007510  104401  001213                 TYPE    ,$CRLF
 2660  007514  000207                         RTS     PC
 2661
 2662                                  ;ROUTINE TO INCREMENT $SOFT
 2663                                  ;
 2664                                  ;NOTE:  $SOFT WILL NOT BE INCREMENTED BEYOND 9999 (10)
 2665
 2666  007516  026027  000052  023417  INCSOF: CMP     $SOFT(R0),#9999.  ;IS $SOFT ALREADY AT MAXIMUM ?
 2667  007524  103002                         BHIS    1$              ;BR IF IT IS
 2668  007526  005260  000052                 INC     $SOFT(R0)       ;INCREMENT $SOFT
 2669  007532  000207                 1$:     RTS     PC              ;RETURN
 2670
 2671                                  ;ROUTINE TO INCREMENT $HARD
 2672                                  ;
 2673                                  ;NOTE:  $HARD WILL NOT BE INCREMENTED BEYOND 9999 (10)
 2674
 2675  007534  026027  000054  023417  INCHRD: CMP     $HARD(R0),#9999.  ;IS $HARD ALREADY AT MAXIMUM ?
 2676  007542  103002                         BHIS    1$              ;BR IF IT IS
 2677  007544  005260  000054                 INC     $HARD(R0)       ;INCREMENT $HARD
 2678  007550  000207                 1$:     RTS     PC              ;RETURN
 2679
 2680                                  ;ROUTINE TO INCREMENT $SKI
 2681                                  ;
 2682                                  ;NOTE:  $SKI WILL NOT BE INCREMENTED BEYOND 9999 (10)
 2683
 2684  007552  026027  000056  023417  INCSKI: CMP     $SKI(R0),#9999.  ;IS $SKI ALREADY AT MAXIMUM ?
 2685  007560  103002                         BHIS    1$              ;BR IF IT IS
 2686  007562  005260  000056                 INC     $SKI(R0)        ;INCREMENT $SKI
 2687  007566  000207                 1$:     RTS     PC              ;RETURN
 2688
 2689                                  ;ROUTINE TO INCREMENT $MISPO
 2690                                  ;
 2691                                  ;NOTE:  $MISPO WILL NOT BE INCREMENTED BEYOND 9999 (10)
 2692
 2693  007570  026027  000060  023417  INCMIS: CMP     $MISPO(R0),#9999.  ;IS $MISPO ALREADY AT MAXIMUM ?
 2694  007576  103002                         BHIS    1$              ;BR IF IT IS
 2695  007600  005260  000060                 INC     $MISPO(R0)      ;INCREMENT $MISPO
 2696  007604  000207                 1$:     RTS     PC              ;RETURN
 2697
 2698                                  ;ROUTINE TO INCREMENT $TOTAL
 2699                                  ;
 2700                                  ;NOTE:  $TOTAL WILL NOT BE INCREMENTED BEYOND 9999 (10)
 2701
 2702  007606  026027  000050  023417  INCTOT: CMP     $TOTAL(R0),#9999.  ;IS $TOTAL ALREADY AT MAXIMUM ?
 2703  007614  103002                         BHIS    1$              ;BR IF IT IS
 2704  007616  005260  000050                 INC     $TOTAL(R0)      ;INCREMENT $TOTAL
 2705  007622  000207                 1$:     RTS     PC              ;RETURN
 2706
 2707
```

```
                                        ;ROUTINE TO TYPE THE TIME
                                        
007624  005737  001236          $TIME:  TST     CLKFLG          ;CLOCK ON THE SYSTEM ?
007630  001006                          BNE     1$              ;BR IF NOT
007632  104401  010120                  TYPE    ,HOUR           ;TYPE THE HOURS
007636  104401  010124                  TYPE    ,MINUTE         ;TYPE THE MINUTES
007642  104401  010130                  TYPE    ,SECOND         ;TYPE THE SECONDS
007646  000207                  1$:     RTS     PC
                                        
                                        ;CLOCK HANDLER ROUTINE
                                        
007650  010046                  CLOCK:  MOV     R0,-(SP)        ;SAVE R0
007652  005237  010134                  INC     SIXTEE          ;INCREMENT THE 1/60 SECOND COUNTER
007656  001076                          BNE     1$              ;BR IF A SECOND NOT COUNTED
007660  013737  001240  010134          MOV     HZ,SIXTEE       ;RESTORE THE VALUE
007666  005437  010134                  NEG     SIXTEE          ;MAKE IT INTO 2'S COMP
007672  105237  010131                  INCB    SECOND+1        ;INCREMENT THE SECONDS
007676  122737  000072  010131          CMPB    #72,SECOND+1    ;SEE IF COUNTER AT MAX
007704  001063                          BNE     1$              ;BR IF NOT
007706  112737  000060  010131          MOVB    #60,SECOND+1    ;RESTORE VALUE
007714  105237  010130                  INCB    SECOND          ;INCREMENT UPPER
007720  122737  000066  010130          CMPB    #66,SECOND      ;UPPER AT MAX
007726  001052                          BNE     1$              ;BR IF NOT
007730  112737  000060  010130          MOVB    #60,SECOND      ;RESET UPPER
007736  105237  001313                  INCB    INTRVL+1        ;INCREMENT STATISTICS TYPEOUT INTERVAL
007742  105237  010125                  INCB    MINUTE+1        ;INCREMENT MINUTE
007746  122737  000072  010125          CMPB    #72,MINUTE+1    ;AT MAX ?
007754  001037                          BNE     1$              ;BR IF NOT
007756  112737  000060  010125          MOVB    #60,MINUTE+1    ;RESET LOWER
007764  105237  010124                  INCB    MINUTE          ;INCREMENT UPPER HALF
007770  122737  000066  010124          CMPB    #66,MINUTE      ;AT MAX ?
007776  001026                          BNE     1$              ;BR IF NOT
010000  112737  000060  010124          MOVB    #60,MINUTE      ;RESET UPPER
010006  105237  010121                  INCB    HOUR+1          ;INCREMENT HOURS
010012  022737  035071  010120          CMP     #35071,HOUR     ;AT MAX (99) ?
010020  001412                          BEQ     2$              ;BR IF IT IS
010022  122737  000072  010121          CMPB    #72,HOUR+1      ;AT MAX ?
010030  001011                          BNE     1$              ;BR IF NOT
010032  112737  000060  010121          MOVB    #60,HOUR+1      ;RESET VALUE
010040  105237  010120                  INCB    HOUR            ;INCREMENT UPPER
010044  000403                          BR      1$
010046  012737  030060  010120  2$:     MOV     #30060,HOUR     ;RESET HOUR FIELD
010054  012746  000021          1$:     MOV     #17.,-(SP)      ;17 MS ON THE STACK
010060  004737  023350                  JSR     PC,RPTMR        ;DRIVER TIMER ROUTINE
010064  105737  001312                  TSTB    INTRVL          ;DISPLAY THE PERFORMANCE SUMMARY ?
010070  001411                          BEQ     3$              ;BR IF NOT
010072  123737  001312  001313          CMPB    INTRVL,INTRVL+1 ;DISPLAY INTERVAL FINISHED ?
010100  001005                          BNE     3$              ;BR IF NOT
010102  105037  001313                  CLRB    INTRVL+1        ;RESTORE THE INTERVAL
010106  012737  177777  001250          MOV     #-1,STATIN      ;SET ERROR RATE DATA DISPLAY FLAG
010114  012600                  3$:     MOV     (SP)+,R0        ;RESTORE R0
010116  000002                          RTI
                                        
010120      060     060         HOUR:   .BYTE   60,60           ;HOURS
010122      072     000                 .BYTE   72,0            ;':' & TERMINATOR
010124      060     060         MINUTE: .BYTE   60,60           ;MINUTES
```

```
 2764  010126      072      000                    .BYTE   72.0              ;':' & TERMINATOR
 2765  010130      060      060          SECOND:   .BYTE   60.60             ;SECONDS
 2766  010132      040      000                    .BYTE   40.0              ;SPACE, TERMINATOR
 2767                                               .EVEN
 2768
 2769  010134  000000                     SIXTEE:   .WORD   0                ;1/60TH OR 1/50TH OF A SECOND COUNTER
 2770
 2771                                     ;COMMAND DECODE ROUTINE
 2772
 2773  010136  104412                     KSR:      SAVREG                   ;SAVE R0-R5
 2774  010140  012746  000200                       MOV     #PR4,-(SP)       ;RETURN AT PRIORITY 4
 2775  010144  012746  010220                       MOV     #KSR1,-(SP)      ;DUMMY INTERRUPT RETURN
 2776  010150  017746  170772                       MOV     @$TKB,-(SP)      ;GET THE CHARACTER FROM THE KEYBOARD
 2777  010154  042716  177600                       BIC     #^C177,(SP)      ;CLEAR THE JUNK
 2778  010160  021627  000007                       CMP     (SP),#7          ;'CONTROL G' ?
 2779  010164  001010                             BNE     1$               ;BR IF NOT
 2780  010166  022737  000176  001140              CMP     #SWREG,SWR       ;SOFTWARE SWITCH REGISTER ?
 2781  010174  001004                             BNE     1$               ;BR IF NOT ACTIVE
 2782  010176  012766  010410  000002              MOV     #KSR2,2(SP)      ;CHANGE THE RETURN ADDRESS
 2783  010204  000403                             BR      2$               ;CONTINUE
 2784  010206  012737  017016  000060  1$:       MOV     #$TKSRV,TKVEC    ;CHANGE KEYBOARD VECTOR ADDRESS
 2785  010214  000137  017022            2$:     JMP     $TKSRV+4         ;PROCESS THE CHARACTER JUST ENTERED
 2786  010220  104411                     KSR1:   RDLIN                    ;READ THE KEYBOARD
 2787  010222  012737  010136  000060              MOV     #KSR,TKVEC       ;CHANGE THE KEYBOARD VECTOR
 2788  010230  012605                             MOV     (SP)+,R5         ;BUFFER ADDRESS
 2789  010232  005205                             INC     R5               ;POINT TO DRIVE NUMBER
 2790  010234  104401  001213                     TYPE    ,$CRLF           ;CR-LF
 2791  010240  122715  000101                     CMPB    #'A,(R5)         ;EQ TO AN 'A'
 2792  010244  001410                             BEQ     1$               ;BR IF IT IS
 2793  010246  121527  000067                     CMPB    (R5),#67         ;UNIT NUMBER GREATER THAN AN ASCII 7 ?
 2794  010252  101054                             BHI     6$               ;BR IF IT IS
 2795  010254  121527  000060                     CMPB    (R5),#60         ;UNIT NUMBER LESS THAN AN ASCII 0 ?
 2796  010260  103451                             BLO     6$               ;BR IF IT IS
 2797  010262  142715  177770                     BICB    #^C7,(R5)        ;LEAVE ONLY LOWER 3 BITS IF CHAR NOT 'A'
 2798  010266  122765  000124  177777  1$:       CMPB    #'T,-1(R5)       ;EQ TO 'T'
 2799  010274  001003                             BNE     2$               ;BR IF NOT EQ
 2800  010276  004737  010414                     JSR     PC,NEWASN        ;ASSIGN UNIT FOR TEST
 2801  010302  000442                             BR      KSR2             ;EXIT
 2802  010304  122765  000104  177777  2$:       CMPB    #'D,-1(R5)       ;EQ TO 'D' ?
 2803  010312  001003                             BNE     3$               ;BR IF NOT EQ
 2804  010314  004737  010654                     JSR     PC,DEASGN        ;DEASSIGN UNIT
 2805  010320  000433                             BR      KSR2             ;EXIT
 2806  010322  122765  000123  177777  3$:       CMPB    #'S,-1(R5)       ;EQ TO 'S'
 2807  010330  001003                             BNE     4$               ;BR IF NOT EQ
 2808  010332  004737  010762                     JSR     PC,SCMND         ;TYPE STATISTICS
 2809  010336  000424                             BR      KSR2             ;EXIT
 2810  010340  122765  000127  177777  4$:       CMPB    #'W,-1(R5)       ;EQ TO 'W'
 2811  010346  001007                             BNE     5$               ;BR IF NOT EQ
 2812  010350  032777  000001  170562              BIT     #SW0,@SWR        ;IS SWITCH 0 SET ?
 2813  010356  001012                             BNE     6$               ;BR IF SET, CAN'T DO 'W' COMMAND
 2814  010360  004737  011064                     JSR     PC,DATAPK        ;WRITE A DATA PACK
 2815  010364  000411                             BR      KSR2             ;EXIT
 2816  010366  122765  000122  177777  5$:       CMPB    #'R,-1(R5)       ;EQ TO 'R' ?
 2817  010374  001003                             BNE     6$               ;BR IF NOT EQ
 2818  010376  004737  011076                     JSR     PC,REDAPK        ;READ A DATA PACK
 2819  010402  000402                             BR      KSR2             ;EXIT
```

```
2820  010404  104401  033136          6$:     TYPE    ,INVLD          ;TYPE 'INVALID COMMAND' MESSAGE
2821  010410  104413                  KSR2:   RESREG                  ;RESTORE R0-R5
2822  010412  000002                          RTI                     ;RETURN
2823
2824                                  ;ROUTINE TO ASSIGN A DRIVE ('T' COMMAND)
2825
2826  010414  005037  001252          NEWASN: CLR     PACK            ;CLEAR 'W' COMMAND INDICATOR
2827  010420  005004                  ASGN0:  CLR     R4              ;R4 IS TABLE INDEX
2828  010422  012703  000010                  MOV     #8.,R3          ;SET INITIAL COUNT TO 8
2829  010426  122715  000101                  CMPB    #101,(R5)       ;ASSIGN ALL UNITS?
2830  010432  001417                          BEQ     ASGN2           ;BR IF ALL UNITS
2831  010434  111504                  ASGN1:  MOVB    (R5),R4         ;PUT UNIT # IN R4
2832  010436  012737  032557  011130          MOV     #UNTASN,ASNMSG  ;'UNIT ASSIGNED' MESSAGE ADDR
2833  010444  012703  000001                  MOV     #1,R3           ;RELOAD R3 FOR 1 UNIT
2834  010450  136437  021274  025450          BITB    ATABIT(R4),ASNLST  ;UNIT ALREADY ASSIGNED ?
2835  010456  001002                          BNE     1$              ;BR IF IT IS
2836  010460  004737  010514                  JSR     PC,ASGN3        ;SEE IF UNIT ON THE SYSTEM
2837  010464  000137  011110          1$:     JMP     ASNERR          ;RETURN HERE IF DRIVE NOT AVAIL
2838  010470  000207                          RTS     PC              ;EXIT
2839  010472  004737  010514          ASGN2:  JSR     PC,ASGN3        ;ASSIGN ALL UNASSIGNED, AVAIL UNITS
2840  010476  000137  010504                  JMP     1$              ;UNIT NOT ON SYSTEM
2841  010502  000207                          RTS     PC              ;EXIT
2842  010504  012746  010476          1$:     MOV     #ASGN2+4,-(SP)  ;PUT RETURN ADDRESS ON THE STACK
2843  010510  000137  010616                  JMP     ASGN4           ;LOOK FOR MORE UNITS
2844  010514  136437  021274  025450  ASGN3:  BITB    ATABIT(R4),ASNLST  ;UNIT ALREADY ASSIGNED ?
2845  010522  001035                          BNE     ASGN4           ;BR IF IT IS
2846  010524  005737  021114          1$:     TST     TRNSWT          ;DATA TRANSFER UNDER WAY ?
2847  010530  001375                          BNE     1$              ;BR IF IT IS
2848  010532  110437  025404                  MOVB    R4,GENDPB       ;DRIVE NUMBER
2849  010536  004737  015344                  JSR     PC,RECAL0       ;RECALIBRATE DRIVE
2850  010542  105764  021102                  TSTB    DRVSTA(R4)      ;DRIVE AVAILABLE?
2851  010546  001432                          BEQ     ASGN6           ;BR IF DRIVE OFFLINE
2852  010550  100435                          BMI     ASGN7           ;BR DRIVE NOT AVAILABLE
2853  010552  006304                          ASL     R4              ;MAKE R4 INTO WORD INDEX
2854  010554  016464  025726  025516          MOV     BLKADR(R4),NEWUNT(R4) ;DPB ADDRESS
2855  010562  016400  025726                  MOV     BLKADR(R4),R0   ;PUT BLOCK'S ADDR INTO R0
2856  010566  004737  011140                  JSR     PC,CLRDPB       ;CLEAR BLOCK FOR UNIT JUST ASSIGNED
2857  010572  012760  000001  000062          MOV     #1,$PASSC(R0)   ;PRESET PASS COUNT TO 1
2858  010600  005737  001252                  TST     PACK            ;WRITE DATA PACK ?
2859  010604  001403                          BEQ     2$              ;BR IF NOT
2860  010606  113760  001252  000022          MOVB    PACK,$PACK(R0)  ;SET READ/WRITE DATA PACK INDICATOR
2861  010614  006204                  2$:     ASR     R4              ;RESTORE UNIT ADDRESS
2862  010616  005303                  ASGN4:  DEC     R3              ;DEC UNIT COUNT
2863  010620  001402                          BEQ     ASGN5           ;BR IF FINISHED
2864  010622  005204                          INC     R4              ;INCREMENT UNIT NUMBER
2865  010624  000733                          BR      ASGN3           ;CONTINUE
2866  010626  062716  000004          ASGN5:  ADD     #4,(SP)         ;INCREMENT RETURN
2867  010632  000407                          BR      ASGN8           ;EXIT
2868  010634  012737  032514  011130  ASGN6:  MOV     #UNTOFF,ASNMSG  ;'OFFLINE' MESSAGE ADDRESS
2869  010642  000403                          BR      ASGN8           ;EXIT
2870  010644  012737  032605  011130  ASGN7:  MOV     #NOTSAF,ASNMSG  ;ADDR OF 'DRIVE UNSAFE' MESSAGE
2871  010652  000207                  ASGN8:  RTS     PC              ;RETURN
2872
2873                                  ;ROUTINE TO DEASSIGN A UNIT ('D' COMMAND)
2874
2875  010654  005004                  DEASGN: CLR     R4
```

```
2876  010656  122715  000101            CMPB    #101,(R5)          ;DEASSIGN ALL UNITS ?
2877  010662  001434                    BEQ     5$                 ;BR IF YES
2878  010664  012703  000001            MOV     #1,R3              ;SET R3 FOR ONE UNIT
2879  010670  111504                    MOVB    (R5),R4            ;GET UNIT NUMBER
2880  010672  136437  021274  025450 1$: BITB    ATABIT(R4),ASNLST  ;UNIT ASSIGNED ?
2881  010700  001414                    BEQ     3$                 ;BR IF NOT
2882  010702  146437  021274  025450    BICB    ATABIT(R4),ASNLST  ;DELETE THE UNIT FROM THE ASSIGNED LIST
2883  010710  006304                    ASL     R4                 ;MAKE ADDR INTO A WORD INDEX
2884  010712  016464  025726  025474    MOV     BLKADR(R4),DUNIT(R4)  ;PUT ADDRESS IN DEASSIGN LIST
2885  010720  006204                    ASR     R4
2886  010722  005303            2$:     DEC     R3                 ;ANY MORE UNITS ?
2887  010724  001412                    BEQ     4$                 ;BR IF NOT
2888  010726  005204                    INC     R4
2889  010730  000760                    BR      1$
2890  010732  122715  000101      3$:   CMPB    #101,(R5)          ;DEASSIGN ALL UNITS ?
2891  010736  001771                    BEQ     2$                 ;BR IF YES
2892  010740  012737  032535  011130    MOV     #UNTNOT,ASNMSG     ;ADDR OF 'NOT ASSIGNED' MESSAGE
2893  010746  004737  011110            JSR     PC,ASNERR          ;REPORT IT
2894  010752  000207            4$:     RTS     PC
2895  010754  012703  000010      5$:   MOV     #8.,R3             ;SET UNIT COUNT TO 8
2896  010760  000744                    BR      1$
2897
2898                              ;ROUTINE TO TYPE UNIT PERFORMANCE SUMMARY  ('S' COMMAND)
2899
2900  010762  005004            SCMND:  CLR     R4
2901  010764  122715  000101            CMPB    #101,(R5)          ;ALL STATISTICS ?
2902  010770  001421                    BEQ     2$                 ;BR IF YES
2903  010772  111504                    MOVB    (R5),R4            ;GET UNIT #
2904  010774  136437  021274  025450    BITB    ATABIT(R4),ASNLST  ;SEE IF UNIT ASSIGNED
2905  011002  001406                    BEQ     1$                 ;BR IF NOT
2906  011004  006304                    ASL     R4                 ;MAKE UNIT ADDR INTO WORD INDEX
2907  011006  016400  025726            MOV     BLKADR(R4),R0      ;ADDR OF BLOCK
2908  011012  004737  007230            JSR     PC,TYPEST          ;TYPE DRIVE STATISTICS
2909  011016  000421                    BR      5$                 ;EXIT
2910  011020  012737  032535  011130 1$: MOV     #UNTNOT,ASNMSG     ;ADDR OF 'NOT ASSIGNED' MSG
2911  011026  004737  011110            JSR     PC,ASNERR          ;TYPE ERROR MESSAGE
2912  011032  000413                    BR      5$                 ;EXIT
2913  011034  012703  000010      2$:   MOV     #8.,R3             ;UNIT COUNT
2914  011040  105737  025450      3$:   TSTB    ASNLST             ;SEE IF ANY UNIT ASSIGNED
2915  011044  001004                    BNE     4$                 ;BR IF ANY ARE
2916  011046  005204                    INC     R4                 ;INCREMENT UNIT ADDRESS
2917  011050  005303                    DEC     R3                 ;DECREMENT COUNTER
2918  011052  001372                    BNE     3$                 ;MORE TO CHECK ?
2919  011054  000402                    BR      5$                 ;NONE ASSIGNED, RETURN
2920  011056  004737  007116      4$:   JSR     PC,STATPR          ;TYPE ALL STATISTICS
2921  011062  000207            5$:     RTS     PC
2922
2923                              ;ROUTINE TO WRITE A DATA PACK ('W' COMMAND)
2924
2925  011064  012737  177777  001252 DATAPK: MOV     #-1,PACK           ;SET THE 'W' COMMAND INDICATOR
2926  011072  000137  010420            JMP     ASGNO              ;ASSIGN REQUESTED UNIT
2927
2928
2929                              ;ROUTINE TO READ A DATA PACK ('R' COMMAND)
2930
2931  011076  012737  000001  001252 REDAPK: MOV     #1,PACK            ;SET THE 'READ' INDICATOR
```

```
 2932  011104  000137  010420                 JMP     ASGNO           ;ASSIGN THE REQUESTED UNIT
 2933
 2934                                 ;TYPEOUT ASSIGN/DEASSIGN ERROR MESSAGE
 2935
 2936  011110  104401  033134         ASNERR: TYPE    ,QUES           ;QUESTION MARK
 2937  011114  104401  032505                 TYPE    ,UNTMSG         ;TYPE 'UNIT'
 2938  011120  010446                         MOV     R4,-(SP)        ;DRIVE NUMBER
 2939  011122  104403                         TYPOS                   ;TYPE IT
 2940  011124     001     000                 .BYTE   1,0
 2941  011126  104401                         TYPE                    ;TYPE SPECIFIC MESSAGE
 2942  011130  000000                 ASNMSG: .WORD   0               ;MESSAGE ADDRESS
 2943  011132  104401  001213                 TYPE    ,$CRLF
 2944  011136  000207                         RTS     PC
 2945
 2946                                 ;ROUTINE TO CLEAR THE DPB FOR THE ASSIGNED UNIT
 2947
 2948  011140                         CLRDPB:
 2949  011140  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 2950  011142  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 2951  011144  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 2952  011146  010004                         MOV     R0,R4           ;GET THE DPB ADDRESS
 2953  011150  062704  000002                 ADD     #2,R4           ;ADDRESS OF FIRST LOCN TO BE CLEARED
 2954  011154  012703  000005                 MOV     #5,R3           ;NUMBER OF LOCNS TO BE CLEARED
 2955  011160  005024         1$:     CLR     (R4)+           ;CLEAR THE LOCATION
 2956  011162  005303                 DEC     R3              ;DECREMENT THE COUNTER
 2957  011164  001375                 BNE     1$              ;BR IF NOT FINISHED
 2958  011166  062704  000002                 ADD     #2,R4           ;MOVE THE ADDRESS PAST THE 'REG' ADDR
 2959  011172  012703  000074                 MOV     #$SILO-$REG,R3  ;NUMBER OF LOCNS TO BE CLEARED
 2960  011176  005024         2$:     CLR     (R4)+           ;CLEAR
 2961  011200  162703  000002                 SUB     #2,R3           ;DECREMENT THE LOCN COUNTER
 2962  011204  001374                 BNE     2$              ;BR IF NOT FINISHED
 2963  011206  012605                 MOV     (SP)+,R5        ;;POP STACK INTO R5
 2964  011210  012604                 MOV     (SP)+,R4        ;;POP STACK INTO R4
 2965  011212  012603                 MOV     (SP)+,R3        ;;POP STACK INTO R3
 2966  011214  000207                 RTS     PC
 2967
 2968                                 ;DEASSIGN A DRIVE IF A FATAL ERROR OCCURS
 2969
 2970  011216  011446         DROP:   MOV     (R4),-(SP)      ;SAVE R4
 2971  011220  005004                 CLR     R4              ;CLEAR R4 FOR UNIT NUMBER
 2972  011222  111004                 MOVB    (R0),R4         ;MOVE UNIT NUMBER TO R4
 2973  011224  146437  021274  025450         BICB    ATABIT(R4),ASNLST  ;REMOVE UNIT FROM ASSIGNED LIST
 2974  011232  006304                 ASL     R4              ;MAKE UNIT NUMBER INTO A TABLE INDEX
 2975  011234  010064  025474                 MOV     R0,DUNIT(R4)    ;PUT UNIT IN DROP LIST
 2976  011240  104401  001213                 TYPE    ,$CRLF          ;CR-LF
 2977  011244  104401  001213                 TYPE    ,$CRLF          ;CR-LF
 2978  011250  104401  032771                 TYPE    ,DROPNG         ;TYPE 'DROPPING UNIT'
 2979  011254  104401  033073                 TYPE    ,DRNUM          ;'DRIVE #'
 2980  011260  111046                         MOVB    (R0),-(SP)      ;DRIVE NUMBER
 2981  011262  104403                         TYPOS                   ;TYPE IT
 2982  011264     001     000                 .BYTE   1,0
 2983  011266  104401  001213                 TYPE    ,$CRLF          ;CR-LF
 2984  011272  012604                         MOV     (SP)+,R4        ;RESTORE R4
 2985  011274  000207                         RTS     PC              ;RETURN
 2986
 2987                                 ;ROUTINE TO DEASSIGN A UNIT IF ERRORS BECOMES EXCESSIVE
```

```
 2988
 2989  011276  032777  000020  167634  ABNRML: BIT     #SW04,@SWR      ;SEE IF SWITCH 4 SET
 2990  011304  001006                          BNE     1$              ;BR IF IT'S SET
 2991  011306  023760  001310  000050          CMP     MAXER,$TOTAL(R0)  ;CHECK TOTAL ERROR VALUE
 2992  011314  101002                          BHI     1$              ;BR IF NOT AT MAXIMUM
 2993  011316  000137  011216                  JMP     DROP            ;DEASSIGN THE DRIVE
 2994  011322  000207                  1$:     RTS     PC              ;RETURN
 2995
 2996                                          ;ROUTINE TO CHECK FOR END OF PASS AND END OF TEST
 2997
 2998  011324  005737  001316          NRML:   TST     SEKMOD          ;END OF PASS DETERMINED BY SEEKS OR WORDS ?
 2999  011330  001012                          BNE     NRML1           ;BR IF SEEKS
 3000  011332  026037  000046  001276          CMP     $READ+2(R0),ENDCON+2  ;CHECK MSW OF WORDS READ COUNT
 3001  011340  101017                          BHI     NRML2           ;BR IF MSW GREATER THAN LIMIT
 3002  011342  103405                          BLO     NRML1           ;BR IF MSW LESS THAN LIMIT
 3003  011344  026037  000044  001274          CMP     $READ(R0),ENDCON  ;CHECK LSW AGAINST LIMIT
 3004  011352  103012                          BHIS    NRML2           ;BR IF EQUAL OR GREATER
 3005  011354  000504                          BR      NRMLX           ;EXIT
 3006  011356  026037  000042  001302  NRML1: CMP     $POSIT+2(R0),ENDSEK+2  ;CHECK MSW OF SEEK COUNT
 3007  011364  101005                          BHI     NRML2           ;BR IF MSW GREATER THAN LIMIT
 3008  011366  103477                          BLO     NRMLX           ;EXIT IF MSW LESS THAN LIMIT
 3009  011370  026037  000040  001300          CMP     $POSIT(R0),ENDSEK  ;CHECK LSW OF SEEK COUNT
 3010  011376  103473                          BLO     NRMLX           ;EXIT IF LSW LESS THAN LIMIT
 3011  011400  104401  001213          NRML2:  TYPE    ,$CRLF          ;CR-LF
 3012  011404  104401  033025                  TYPE    ,ENDPAS         ;END OF PASS FOR THE UNIT
 3013  011410  016046  000062                  MOV     $PASSC(R0),-(SP)  ;PUT PASS COUNT ON THE STACK
 3014  011414  104405                          TYPDS                   ;CONVERT PASS COUNT TO DECIMAL AND TYPE IT
 3015  011416  032777  000020  167514          BIT     #SW04,@SWR      ;SWITCH 4 SET ?
 3016  011424  001016                          BNE     1$              ;BR IF SET
 3017  011426  026037  000062  001304          CMP     $PASSC(R0),PASCNT  ;SEE IF AT END OF TEST
 3018  011434  103412                          BLO     1$              ;BR IF NOT
 3019  011436  104401  033041                  TYPE    ,ENDTST         ;TYPE 'END OF TEST'
 3020  011442  104401  033073                  TYPE    ,DRNUM          ;'DRIVE #'
 3021  011446  111046                          MOVB    (R0),-(SP)      ;DRIVE NUMBER
 3022  011450  104403                          TYPOS                   ;TYPE IT
 3023  011452     001     000                  .BYTE   1,0
 3024  011454  104401  001213                  TYPE    ,$CRLF          ;CR-LF
 3025  011460  000430                          BR      3$              ;DEASSIGN THE DRIVE
 3026  011462  104401  033073          1$:     TYPE    ,DRNUM          ;'DRIVE #'
 3027  011466  111046                          MOVB    (R0),-(SP)      ;DRIVE NUMBER
 3028  011470  104403                          TYPOS                   ;TYPE IT
 3029  011472     001     000                  .BYTE   1,0
 3030  011474  104401  001213                  TYPE    ,$CRLF          ;CR-LF
 3031  011500  004737  007230                  JSR     PC,TYPEST       ;TYPE THE UNIT'S STATISTICS
 3032  011504  010346                          MOV     R3,-(SP)        ;SAVE R3
 3033  011506  010446                          MOV     R4,-(SP)        ;SAVE R4
 3034  011510  010004                          MOV     R0,R4           ;UNIT'S BLOCK ADDRESS
 3035  011512  062704  000034                  ADD     #$OPERC,R4      ;ADD THE STARTING ADDR OF SECTIONS TO CLEAR
 3036  011516  012703  000006                  MOV     #6,R3           ;NUMBER OF LOCNS TO BE CLEARED
 3037                                          ;(ERROR COUNTERS NOT CLEARED)
 3038  011522  005024                  2$:     CLR     (R4)+           ;CLEAR THE LOCN
 3039  011524  005303                          DEC     R3              ;DECREMENT THE LOCATION COUNTER
 3040  011526  001375                          BNE     2$              ;BR IF MORE TO GO
 3041  011530  012604                          MOV     (SP)+,R4        ;RESTORE R4
 3042  011532  012603                          MOV     (SP)+,R3        ;RESTORE R3
 3043  011534  005260  000062                  INC     $PASSC(R0)      ;INCREMENT THE PASS COUNT
```

```
3044  011540  000412                          BR      NRMLX           ;EXIT
3045  011542  104401  001213          3$:     TYPE    ,$CRLF
3046  011546  005004                          CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
3047  011550  111004                          MOVB    (R0),R4         ;MOVE DRIVE NUMBER
3048  011552  146437  021274  025450          BICB    ATABIT(R4),ASNLST  ;DELETE DRIVE FROM ASSIGNED LIST
3049  011560  006304                          ASL     R4              ;MAKE DRIVE NUMBER INTO TABLE INDEX
3050  011562  010064  025474                  MOV     R0,DUNIT(R4)    ;PUT BLOCK ADDRESS INTO DROP LIST
3051  011566  000207                  NRMLX:  RTS     PC              ;RETURN
3052
3053                                  ;ROUTINE TO GET THE REMAINDER OF THE RANDOM NUMBER
3054
3055  011570  013746  020610          GETREM: MOV     $HINUM,-(SP)    ;LOWER DIVIDEND
3056  011574  005046                          CLR     -(SP)           ;UPPER DIVIDENT
3057  011576  010546                          MOV     R5,-(SP)        ;PUT DIVISOR ON THE STACK
3058  011600  004737  020176                  JSR     PC,$DIV
3059  011604  012605                          MOV     (SP)+,R5        ;PUT THE REMAINDER INTO R5
3060  011606  005726                          TST     (SP)+           ;ADJUST THE STACK POINTER
3061  011610  000207                          RTS     PC
3062
3063                                  ;ROUTINE TO PEPLACE LEADING ZEROS IN A NUMERIC STRING WITH SPACES
3064
3065  011612  012737  000004  011732  $RPZR8: MOV     #4,RPZRC        ;SETUP TO TYPE 8 SIGNIFICANT DIGITS
3066  011620  000412                          BR      RPZERO          ;TYPE THE VALUE
3067  011622  012737  000006  011732  $RPZR4: MOV     #6,RPZRC        ;SETUP TO TYPE 4 SIGNIFICANT DIGITS
3068  011630  000406                          BR      RPZERO          ;TYPE THE VALUE
3069  011632  012737  000007  011732  $RPZR5: MOV     #7,RPZRC        ;SETUP TO TYPE 3 SIGNIFICANT DIGITS
3070  011640  000402                          BR      RPZERO          ;TYPE THE VALUE
3071  011642  005037  011732          $RPZR0: CLR     RPZRC           ;SETUP TO TYPE 10 SIGNIFICANT DIGITS
3072  011646  010046                  RPZERO: MOV     R0,-(SP)        ;SAVE R0
3073  011650  016600  000004                  MOV     4(SP),R0        ;ADDRESS OF NUMBER TO R0
3074  011654  122710  000060          1$:     CMPB    #'0,(R0)        ;BYTE EQUAL TO ASCII '0' ?
3075  011660  001004                          BNE     2$              ;BR IF NOT
3076  011662  112710  000040                  MOVB    #40,(R0)        ;REPLACE THE ZERO WITH A SPACE
3077  011666  005200                          INC     R0              ;INCREMENT THE BYTE ADDRESS
3078  011670  000771                          BR      1$              ;GO BACK AND LOOK FOR MORE LEADING ZEROS
3079  011672  105710                  2$:     TSTB    (R0)            ;SEE IF ZERO BYTE TERMINATOR
3080  011674  001003                          BNE     3$              ;BR IF NOT
3081  011676  005300                          DEC     R0              ;BACKUP STRING POINTER
3082  011700  112710  000060                  MOVB    #'0,(R0)        ;PUT A ZERO BACK IN
3083  011704  016637  000004  011722  3$:     MOV     4(SP),4$        ;PUT ADDRESS IN LOCATION FOR TYPEOUT
3084  011712  063737  011732  011722          ADD     RPZRC,4$        ;BEGINNING OF SIGNIFICANT DIGITS
3085  011720  104401                          TYPE                    ;TYPE THE NUMBER
3086  011722  000000                  4$:     .WORD   0               ;ADDRESS OF NUMBER
3087  011724  012600                          MOV     (SP)+,R0        ;RESTORE R0
3088  011726  012616                          MOV     (SP)+,(SP)      ;MOVE RETURN ADDRESS
3089  011730  000207                          RTS     PC              ;RETURN
3090
3091  011732  000000                  RPZRC:  .WORD   0               ;OFFSET FOR CONVERTED DIGITS HERE
3092
3093                                  ;THIS ROUTINE IS USED TO CHECK IF AN ASCII CHARACTER IS A DIGIT
3094                                  ;BETWEEN 0 AND 7.
3095                                  ;
3096                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3097                                  ;       JSR     R0,CK.OCT       ;CHECK THE CHARACTER
3098                                  ;       RETURN1                 ;CHARACTER IS IN R2 AS A
3099                                  ;                               ;OCTAL DIGIT
```

```
3100                                   ;       RETURN2                 ;CHARACTER IS NOT BETWEEN 0-7
3101
3102  011734  121127  000060          CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
3103  011740  103407                          BLO     1$              ;YES -- BRANCH
3104  011742  121127  000067                  CMPB    (R1),#'7        ;GREATER THAN SEVEN?
3105  011746  101004                          BHI     1$              ;YES -- BRANCH
3106  011750  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
3107  011752  042702  177770                  BIC     #^C7,R2         ;STRIP AWAY THE ASCII
3108  011756  000401                          BR      2$              ;BYPASS RETURN ADJUST
3109  011760  005720                  1$:     TST     (R0)+           ;ADJUST FOR RETURN
3110  011762  000200                  2$:     RTS     R0              ;RETURN
3111
3112                                   ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
3113                                   ;DETERMINE WHAT IT IS.
3114                                   ;
3115                                   ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3116                                   ;       JSR     R0,CK.CHR       ;CHECK CHARACTER
3117                                   ;       RETURN  ADR1            ;UNKNOWN CHARACTER
3118                                   ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)-ADR+1
3119                                   ;       RETURN  ADR3            ;PERIOD * (R1)=ADR+1
3120
3121  011764  105711                  CK.CHR: TSTB    (R1)            ;''CARRIAGE RETURN''?
3122  011766  001404                          BEQ     1$              ;YES -- BRANCH
3123  011770  121127  000056                  CMPB    (R1),#'.        ;''PERIOD''?
3124  011774  001003                          BNE     2$              ;NO -- BRANCH
3125  011776  005720                          TST     (R0)+           ;PERIOD
3126  012000  005720                  1$:     TST     (R0)+           ;CARRIAGE RETURN
3127  012002  005201                          INC     R1              ;MOVE POINTER TO NEXT CHARACTER
3128  012004  011000                  2$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
3129  012006  000200                          RTS     R0              ;RETURN
3130
3131
3132                                   ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
3133                                   ;AND FORMS AN OCTAL NUMBER IN R2
3134                                   ;
3135                                   ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
3136                                   ;       JSR     R0,CK.NUM       ;GO FORM THE NUMBER
3137                                   ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
3138                                   ;       RETURN  ADR2            ;''CR''--R2=NUMBER
3139                                   ;       RETURN  ADR3            ;''PERIOD''--R2=NUMBER
3140                                   ;       RETURN  ADR4            ;'CR' WAS FIRST ENTRY
3141                                   ;       RETURN  ADR5            ;'PERIOD' WAS FIRST ENTRY
3142
3143  012010  010346                  CK.NUM: MOV     R3,-(SP)        ;SAVE R3
3144  012012  005003                          CLR     R3              ;START NUMBER AT ZERO
3145  012014  004037  011734                  JSR     R0,CK.OCT       ;OCTAL DIGIT?
3146  012020  000405                          BR      1$              ;YES--BRANCH
3147  012022  004037  011764                  JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
3148  012026  012112                          6$                      ;ILLEGAL CHARACTER
3149  012030  012102                          8$                      ;CARRIAGE RETURN
3150  012032  012100                          7$                      ;''.''
3151  012034  005201                  1$:     INC     R1              ;MOVE TO NEXT CHARACTER
3152  012036  006303                          ASL     R3              ;FOR THE OCTAL NUMBER IN R3
3153  012040  103424                          BCS     6$              ;DON'T LET IT GET TO BIG
3154  012042  006303                          ASL     R3
3155  012044  103422                          BCS     6$
```

```
3156  012046  006303                          ASL     R3
3157  012050  103420                          BCS     6$
3158  012052  060203                          ADD     R2,R3
3159  012054  004037  011734                  JSR     R0,CK.OCT       ;IS THIS AN OCTAL DIGIT?
3160  012060  000765                          BR      1$              ;YES--MAKE IT PART OF THE NUMBER
3161  012062  010302                  2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
3162  012064  005003                          CLR     R3              ;START WITH ZERO INDEX
3163  012066                          3$:
3164  012066  004037  011764                  JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
3165  012072  012112                          6$                      ;ILLEGAL CHARACTER
3166  012074  012106                          5$                      ;CARRIAGE RETURN
3167  012076  012104                          4$                      ;"."
3168  012100  005723                  7$:     TST     (R3)+           ;'PERIOD' ONLY
3169  012102  005723                  8$:     TST     (R3)+           ;'CR' ONLY
3170  012104  005723                  4$:     TST     (R3)+           ;'PERIOD"
3171  012106  005723                  5$:     TST     (R3)+           ;'CR"
3172  012110  060300                          ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
3173  012112  012603                  6$:     MOV     (SP)+,R3        ;RESTORE R3
3174  012114  011000                          MOV     (R0),R0         ;PICKUP EXIT ADDRESS
3175  012116  000200                          RTS     R0              ;RETURN
3176
3177                                  ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
3178                                  ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
3179                                  ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
3180                                  ;CONCERNING THE ERROR.
3181
3182  012120  104412                  TYPERR: SAVREG                  ;SAVE R0-R5
3183  012122  032777  020000  167010          BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS ?
3184  012130  001113                          BNE     20$             ;BR IF INHIBIT
3185  012132  005000                          CLR     R0              ;CLEAR R0 FOR ERROR NUMBER
3186  012134  113700  001114                  MOVB    $ITEMB,R0       ;ERROR NUMBER
3187  012140  005300                          DEC     R0              ;FORM INDEX FOR ERROR TABLE
3188  012142  006300                          ASL     R0
3189  012144  006300                          ASL     R0
3190  012146  006300                          ASL     R0
3191  012150  062700  001346          1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
3192  012154  012037  012174                  MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
3193  012160  001406                          BEQ     3$              ;BRANCH IF THERE ISN'T ONE
3194  012162  104401  001213                  TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
3195  012166  004737  007624                  JSR     PC,$TIME        ;TYPE THE TIME
3196  012172  104401                          TYPE
3197  012174  000000                  2$:     .WORD   0               ;"EM" POINTER GOES HERE
3198  012176  012037  012212          3$:     MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
3199  012202  001404                          BEQ     5$              ;BRANCH IF NONE
3200  012204  104401  001213                  TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3201  012210  104401                          TYPE
3202  012212  000000                  4$:     .WORD   0               ;"DH" POINTER GOES HERE
3203  012214  012001                  5$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
3204  012216  001460                          BEQ     20$             ;BRANCH IF NONE
3205  012220  005005                          CLR     R5              ;SET INDENT SWITCH
3206  012222  012000                          MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
3207  012224  012002                          MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
3208  012226  001451                          BEQ     17$             ;BRANCH IF DH NUMBER IS 0
3209  012230  005105                          COM     R5              ;NO INDENT
3210  012232  104401  001213                  TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3211  012236  112003                  10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
```

```
3212  012240  112004                    MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
3213  012242  006004            11$:    ROR     R4              ;OCTAL OR DECIMAL?
3214  012244  103403                    BCS     12$             ;DECIMAL--BRANCH
3215  012246  013146                    MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3216  012250  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3217  012252  000402                    BR      13$
3218  012254                    12$:
3219  012254  013146                    MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3220  012256  104405                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3221  012260  005303            13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
3222  012262  001403                    BEQ     14$             ;NO--BRANCH
3223  012264  104401  032470            TYPE    ,SPACE2         ;YES--TYPE SEPERATORS
3224  012270  000764                    BR      11$             ;LOOP
3225  012272  005302            14$:    DEC     R2              ;MORE DH'S?
3226  012274  003431                    BLE     20$             ;NO--BRANCH
3227  012276  104401  001213            TYPE    ,$CRLF          ;YES--START A NEW LINE
3228  012302  005760  000002            TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
3229  012306  001404                    BEQ     15$             ;BR IF YES - BYPASS THE INDENT
3230  012310  005105                    COM     R5              ;INDENT?
3231  012312  001002                    BNE     15$             ;NO--BRANCH
3232  012314  104401  032470            TYPE    ,SPACE2         ;YES--TYPE SPACES
3233  012320  012037  012326    15$:    MOV     (R0)+,16$       ;GET NEXT DH
3234  012324  104401                    TYPE                    ;AND TYPE IT
3235  012326  000000            16$:    .WORD   0               ;DH POINTER GOES HERE
3236  012330  005710                    TST     (R0)            ;TYPE A 'DT' ?
3237  012332  001003                    BNE     21$             ;BR IF A 'DT'
3238  012334  062700  000004            ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
3239  012340  000754                    BR      14$             ;SEE IF END OF 'DF' BLOCK
3240  012342  104401  001213    21$:    TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3241  012346  005705                    TST     R5              ;INDENT?
3242  012350  001332                    BNE     10$             ;NO--BRANCH
3243  012352  104401  032470    17$:    TYPE    ,SPACE2         ;YES--TYPE SPACES
3244  012356  000727                    BR      10$             ;LOOP
3245  012360  104413            20$:    RESREG                  ;RESTORE R0-R5
3246  012362  000207                    RTS     PC              ;RETURN
```

```
                                        ;;*****************************************************************
                                        .SBTTL  END OF ORDER PROCESSING
                                        ;;*****************************************************************
                                        ;PROCESS THE ORDER TERMINATION
012364  005760  000016          PROCES: TST     $STATUS(R0)       ;SEE IF RP11 HANDLER SIGNALED AN ERROR
012370  100002                          BPL     1$                ;BR IF NO ERROR DETECTED
012372  000137  013452                  JMP     ERPROC            ;ERROR - PROCESS IT
012376  004737  012424          1$:     JSR     PC,CKERR          ;NO ERROR, CHECK ERROR BITS ANYWAY
012402  004737  012456                  JSR     PC,CKBUS          ;NO ERROR, CHECK BUS ADDR & WC
012406  032777  000002  166524          BIT     #SW01,@SWR        ;COMPARE THE BUFFER ?
012414  001002                          BNE     2$                ;BR IF NOT
012416  004737  012524                  JSR     PC,CMPAR          ;NO ERROR, COMPARE DATA
012422  000207                  2$:     RTS     PC                ;RETURN
                                        ;CHECK ERROR BITS IN 'RPER' & 'RPDS'
012424  032760  015000  000066  CKERR:  BIT     #HNF!SUSI.SUFU,$RPDS(R0)  ;SEE IF ERROR BITS IN RPDS
012432  001003                          BNE     1$                ;BR IF ANY SET
012434  005760  000070                  TST     $RPER(R0)         ;ANY BITS SET IN RPER
012440  001405                          BEQ     2$                ;BR IF NONE SET
012442  104015                  1$:     ERROR   15                ;ERROR BITS SET, NO ERROR SIGNALED
012444  004737  007606                  JSR     PC,INCTOT         ;INCREMENT TOTAL ERROR COUNT
012450  062706  000002                  ADD     #2,SP             ;RETURN TO MAIN ROUTINE
012454  000207                  2$:     RTS     PC                ;RETURN
                                        ;CHECK BUS ADDRESS REGISTER & WORD COUNT REGISTER
012456  016046  000020          CKBUS:  MOV     $WRDL(R0),-(SP)   ;WORD LENGTH
012462  006316                          ASL     (SP)              ;CHANGE INTO BYTE COUNT
012464  066016  000006                  ADD     $BUF(R0),(SP)     ;ADD THE STARTING LOCATION
012470  011637  001124                  MOV     (SP),$GDDAT       ;STORE IN CASE OF ERROR
012474  022660  000076                  CMP     (SP)+,$RPBA(R0)   ;BUFFER ADDRESS PROPER ?
012500  001003                          BNE     1$                ;BR IF NOT CORRECT
012502  005760  000074                  TST     $RPWC(R0)         ;CHECK WORD COUNT
012506  001405                          BEQ     2$                ;BR IF ZERO
012510  104016                  1$:     ERROR   16                ;WORD COUNT OR BUFFER ADDRESS INCORRECT
012512  004737  007606                  JSR     PC,INCTOT         ;INCREMENT TOTAL ERROR COUNT
012516  062706  000002                  ADD     #2,SP             ;RETURN TO MAIN ROUTINE
012522  000207                  2$:     RTS     PC                ;RETURN
                                        ;COMPARE THE BUFFER
012524  122760  000007  000002  CMPAR:  CMPB    #WRTCHK,$COMND(R0)  ;WRITE CHECK ?
012532  001404                          BEQ     1$                ;BR IF WRITE CHECK
012534  132760  000004  000002          BITB    #BIT02,$COMND(R0)  ;READ ORDER ?
012542  001001                          BNE     CMPARD            ;BR IF READ
012544  000207                  1$:     RTS     PC                ;RETURN
012546  016001  000006          CMPARD: MOV     $BUF(R0),R1       ;BUFFER ADDRESS
012552  016037  000020  013144          MOV     $WRDL(R0),CMCNT   ;WORD COUNT TO WORKING LOCATION
```

```
 3303  012560  066037  000074  013144          ADD     $RPWC(R0),CMCNT  ;CALCULATE ACTUAL WORDS TRANSFERED
 3304  012566  016037  000010  013146          MOV     $CYL(R0),CMCYL   ;CYLINDER ADDRESS
 3305  012574  116037  000013  013152          MOVB    $TRK(R0),CMTRK   ;TRACK ADDRESS
 3306  012602  116037  000012  013150          MOVB    $SEC(R0),CMSEC   ;SECTOR ADDRESS
 3307  012610  013737  001314  013142  CMSTR:  MOV     CMPLMT,LIMIT     ;DISPLAY LIMIT
 3308  012616  005237  013142                  INC     LIMIT            ;CONVERT PARAMETER INTO LIMIT VALUE
 3309  012622  010137  013154                  MOV     R1,CMBUF         ;STARTING ADDRESS OF SECTOR BUFFER
 3310  012626  112737  177777  013132          MOVB    #-1,ZROIND       ;CLEAR THE 'ZERO'S' INDICATOR
 3311  012634  105037  013133                  CLRB    FRSTER           ;CLEAR 'FIRST ERROR' INDICATOR
 3312  012640  005037  013140                  CLR     ERCTR            ;CLEAR ERROR COUNTER
 3313  012644  005037  013134                  CLR     SAVER1           ;CLEAR THE R1 SAVE WORD
 3314  012650  005037  013136                  CLR     SAVER5           ;CLEAR THE R5 SAVE WORD
 3315  012654  023727  013144  000400          CMP     CMCNT,#256.      ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
 3316  012662  101003                          BHI     1$               ;BR IF IT IS
 3317  012664  013702  013144                  MOV     CMCNT,R2         ;LESS THAN, USE REMAINING BUFFER
 3318  012670  000402                          BR      2$               ;
 3319  012672  012702  000400           $:     MOV     #256.,R2         ;COMPARE SECTOR
 3320  012676  160237  013144           2$:    SUB     R2,CMCNT         ;DECREMENT WORD COUNT
 3321  012702  004737  013374           CMDAT: JSR     PC,MATCH         ;FIND THE PATTERN
 3322  012706  000403                          BR      2$               ;FOUND A PATTERN
 3323  012710  004737  015030                  JSR     PC,NOMTCH        ;RETURN HERE IF NO MATCH WITH PATTERN MADE
 3324  012714  000456                          BR      9$               ;BYPASS COMPARE ROUTINE
 3325  012716  011405                  2$:     MOV     (R4),R5          ;ADDRESS OF PATTERN ADDRESS IN R4
 3326  012720  012703  000020                  MOV     #20,R3           ;R3 IS PATTERN POS COUNTER
 3327  012724  022125                  3$:     CMP     (R1)+,(R5)+      ;COMPARE BUFFER WITH PATTERN
 3328  012726  001016                          BNE     5$               ;BR IF NOT EQUAL
 3329  012730  005737  013140                  TST     ERCTR            ;ERRORS DETECTED ?
 3330  012734  001406                          BEQ     4$               ;BR IF NO ERRORS
 3331  012736  032777  000010  166174          BIT     #SW3,@SWR        ;SWITCH 3 SET ?
 3332  012744  001402                          BEQ     4$               ;BR IF NOT SET
 3333  012746  004737  013156                  JSR     PC,CMPRT         ;DISPLAY THE WORD
 3334  012752  005302                  4$:     DEC     R2               ;DECREMENT SIZE COUNT
 3335  012754  001434                          BEQ     8$               ;BR WHEN AT END
 3336  012756  005303                          DEC     R3               ;DECREMENT PATT POS COUNT
 3337  012760  001361                          BNE     3$               ;BR IF NOT AT END OF PATT
 3338  012762  000755                          BR      2$               ;RESTART THE PATTERN
 3339  012764  005761  177776          5$:     TST     -2(R1)           ;IS MISCOMPARED CHARACTER=0
 3340  012770  001406                          BEQ     6$               ;BR IF YES
 3341  012772  112737  177777  013132          MOVB    #-1,ZROIND       ;SET NON-ZERO MISCOMPARED IND
 3342  013000  004737  013156                  JSR     PC,CMPRT         ;REPORT ERROR
 3343  013004  000762                          BR      4$               ;CONTINUE COMPARE
 3344  013006  105737  013133          6$:     TSTB    FRSTER           ;FIRST ERROR?
 3345  013012  001007                          BNE     7$               ;BR IF NOT
 3346  013014  105037  013132                  CLRB    ZROIND           ;SET THE ZERO INDICATOR
 3347  013020  010137  013134                  MOV     R1,SAVER1        ;SAVE CURRENT R1
 3348  013024  010537  013136                  MOV     R5,SAVER5        ;SAVE CURRENT R5
 3349  013030  000750                          BR      4$               ;CONTINUE COMPARE
 3350  013032  105737  013132          7$:     TSTB    ZROIND           ;ANY MISCOMPARIONS NOT ZEROS ?
 3351  013036  001745                          BEQ     4$               ;BR IF NONE: ALL ERRORS ARE ZERO
 3352  013040  004737  013156                  JSR     PC,CMPRT         ;REPORT ERROR
 3353  013044  000742                          BR      4$               ;CONTINUE COMPARING
 3354  013046  004737  013352          8$:     JSR     PC,ENDCMP        ;PRINT LAST LINE (IF ERRORS)
 3355  013052  005737  013144          9$:     TST     CMCNT            ;AT END OF BUFFER
 3356  013056  003424                          BLE     CMPRX            ;BR IF AT END
 3357  013060  005237  013150                  INC     CMSEC            ;INCREMENT SECTOR
 3358  013064  023727  013150  000012          CMP     CMSEC,#10.       ;SECTOR GREATER THAN MAX ?
```

```
 3359  013072  103646                          BLO     CMSTR           ;BR IF NOT GREATER THAN MAX
 3360  013074  005037  013150                  CLR     CMSEC           ;CLEAR SECTOR ADDRESS
 3361  013100  005237  013152                  INC     CMTRK           ;INCREMENT TRACK
 3362  013104  023727  013152  000024          CMP     CMTRK,#20.      ;TRACK GREATER THAN MAX ?
 3363  013112  103636                          BLO     CMSTR           ;BR IF NOT GREATER
 3364  013114  005037  013152                  CLR     CMTRK           ;RESET TRACK ADDRESS
 3365  013120  005237  013146                  INC     CMCYL           ;INCREMENT CYLINDER ADDRESS
 3366  013124  000137  012610                  JMP     CMSTR           ;CONTINUE WITH COMPARF
 3367  013130  000207                  CMPRX:  RTS     PC
 3368
 3369  013132     377                  ZROIND: .BYTE   -1              ;ZERO INDICATOR
 3370  013133     000                  FRSTER: .BYTE   0               ;FIRST ERROR INDICATOR
 3371                                                                  ;IF NOT 0, MISCOMPARSION FOUND
 3372  013134  000000                  SAVER1: .WORD   0               ;SAVE R1 HERE
 3373  013136  000000                  SAVER5: .WORD   0               ;SAVE R5 HERE
 3374  013140  000000                  ERCTR:  .WORD   0               ;NUMBER OF ERRORS
 3375  013142  000000                  LIMIT:  .WORD   0               ;DISPLAY LIMIT
 3376  013144  000000                  CMCNT:  .WORD   0               ;WORD COUNT
 3377  013146  000000                  CMCYL:  .WORD   0               ;CYLINDER ADDRESS
 3378  013150  000000                  CMSEC:  .WORD   0               ;SECTOR ADDRESS
 3379  013152  000000                  CMTRK:  .WORD   0               ;TRACK ADDRESS
 3380  013154  000000                  CMBUF:  .WORD   0               ;BEGINNING ADDRESS OF SECTOR
 3381
 3382                                  ;TYPE DATA COMPARE ERRORS
 3383
 3384  013156  105737  013133          CMPRT:  TSTB    FRSTER          ;FIRST ERROR?
 3385  013162  001010                          BNE     1$              ;BR IF NOT
 3386  013164  112737  000012  001114          MOVB    #12,$ITEMB      ;ERROR TABLE INDEX
 3387  013172  004737  012120                  JSR     PC,TYPERR       ;DATA COMPARISON ERROR
 3388  013176  112737  177777  013133          MOVB    #-1,FRSTER      ;SET 'FIRST ERROR' INDICATOR
 3389  013204  005737  013134          1$:     TST     SAVER1          ;SAVED REGISTERS ?
 3390  013210  001420                          BEQ     2$              ;BR IF NONE
 3391  013212  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 3392  013214  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 3393  013216  013701  013134                  MOV     SAVER1,R1       ;DISPLAY SAVED R1
 3394  013222  013705  013136                  MOV     SAVER5,R5       ;DISPLAY SAVED R5
 3395  013226  004737  013264                  JSR     PC,CMPRT1       ;PRINT SAVED VALUES
 3396  013232  005237  013140                  INC     ERCTR           ;INCREMENT THE ERROR COUNTER
 3397  013236  005037  013134                  CLR     SAVER1          ;CLEAR SAVED REGISTER INDICATORS
 3398  013242  005037  013136                  CLR     SAVER5          ;CLEAR THE OTHER ONE
 3399  013246  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 3400  013250  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 3401  013252  004737  013264          2$:     JSR     PC,CMPRT1       ;PRINT REMAINDER OF MESSAGE
 3402  013256  005237  013140                  INC     ERCTR           ;INCREMENT THE ERROR COUNTER
 3403  013262  000207                          RTS     PC              ;RETURN
 3404  013264  005737  013142          CMPRT1. TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
 3405  013270  001403                          BEQ     1$              ;BR IF IT HAS
 3406  013272  005337  013142                  DEC     LIMIT           ;DECREMENT LIMIT COUNTER
 3407  013276  001004                          BNE     2$              ;BR IF NOT AT LIMIT
 3408  013300  032777  000004  165632  1$:     BIT     #SW02,@SWR      ;PRINT ALL DATA COMPARE ERRORS ?
 3409  013306  001420                          BEQ     3$              ;BR IF NOT
 3410  013310  010137  001122          2$:     MOV     R1,$BDADR       ;ADDRESS OF BAD WORD
 3411  013314  162737  000002  001122          SUB     #2,$BDADR       ;ADJUST ADDRESS
 3412  013322  016537  177776  001124          MOV     -2(R5),$GDDAT   ;GOOD DATA
 3413  013330  016137  177776  001126          MOV     -2(R1),$BDDAT   ;BAD DATA
 3414  013336  112737  000013  001114          MOVB    #13,$ITEMB      ;ERROR TABLE INDEX
```

```
3415  013344  004737  012120                  JSR     PC,TYPERR       ;DISPLAY WORD THAT DIDN'T COMPARE
3416  013350  000207                  3$:     RTS     PC              ;RETURN
3417
3418                                  ;LAST LINE OF COMPARE ERROR REPORTING
3419
3420  013352  005737  013140          ENDCMP: TST     ERCTR           ;SEE IF ANY ERRORS
3421  013356  001405                          BEQ     1$              ;BR IF NONE
3422  013360  104014                          ERROR   14              ;NUMBER OF COMPARE ERRORS
3423  013362  004737  007606                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
3424  013366  062706  000002                  ADD     #2,SP           ;RETURN TO MAIN ROUTINE
3425  013372  000207                  1$:     RTS     PC              ;RETURN
3426
3427                                  ;FIND THE CORRECT PATTERN - RETURN WITH ADDRESS OF PATTERN IN R4
3428                                  ;       RETURN +2 IF PATTERN CAN'T BE FOUND
3429
3430  013374  010146                  MATCH:  MOV     R1,-(SP)        ;SAVE R1 ON THE STACK
3431  013376  012704  000044                  MOV     #44,R4          ;PATTERN TABLE INDEX
3432  013402  011601                  1$:     MOV     (SP),R1         ;RELOAD R1
3433  013404  162704  000002                  SUB     #2,R4           ;DECREMENT INDEX
3434  013410  016405  025754                  MOV     STNDAT(R4),R5   ;ADDRESS OF PATTERN ADDRESS
3435  013414  001411                          BEQ     3$              ;BR IF ALL PATTERNS CHECKED AND NO MATCH
3436                                                                  ;FOUND
3437  013416  012703  000004                  MOV     #4,R3           ;NUMBER OF LOCATIONS TO CHECK
3438  013422  022125                  2$:     CMP     (R1)+,(R5)+     ;COMPARE THE BUFFER AGAINST THE PATTERN
3439  013424  001366                          BNE     1$              ;BR IF NOT EQUAL, TRY NEXT PATTERN
3440  013426  005303                          DEC     R3              ;FINISHED CHECKING?
3441  013430  001374                          BNE     2$              ;BR IF NOT FINISHED
3442  013432  062704  025754                  ADD     #STNDAT,R4      ;MAKE PATTERN ADDRESS ABSOLUTE
3443  013436  000403                          BR      4$              ;EXIT
3444  013440  062766  000002  000002  3$:     ADD     #2,2(SP)        ;INCREMENT RETURN ADDRESS
3445  013446  012601                  4$:     MOV     (SP)+,R1        ;RESTORE R1
3446  013450  000207                          RTS     PC              ;RETURN
3447
3448                                  ;ORDER TERMINATED WITH AN ERROR - PROCESS THE ERROR
3449
3450  013452  032760  000200  000016  ERPROC: BIT     #BIT07,$TATUS(R0) ;DONE BIT SET ?
3451  013460  001402                          BEQ     ERPRC1          ;BR IF ORDER DIDN'T COMPLETE NORMALLY
3452  013462  000137  013652                  JMP     DONE            ;PROCESS ERROR WITH 'DONE' BIT SET
3453
3454                                  ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE NOT' BITS
3455
3456  013466  032760  000400  000016  ERPRC1: BIT     #BIT08,$TATUS(R0)  ;WAS A 'RESET' ISSUED ?
3457  013474  001405                          BEQ     1$              ;BR IF NOT
3458  013476  012777  000100  165440          MOV     #BIT06,@$TKS    ;RESTORE TTY KEYBOARD INTERRUPT
3459  013504  004737  006752                  JSR     PC,CKCLK        ;START THE CLOCK
3460  013510  032760  040000  000016  1$:     BIT     #BIT14,$TATUS(R0)  ;UNIT WENT OFFLINE ?
3461  013516  001022                          BNE     OFLIN           ;BR IF IT DID
3462  013520  032760  020000  000016          BIT     #BIT13,$TATUS(R0)  ;BAD COMMAND CODE USED ?
3463  013526  001023                          BNE     BADOP           ;BR IF SO
3464  013530  032760  010000  000016          BIT     #BIT12,$TATUS(R0) ;SEE IF DRIVE WAS UNSAFE
3465  013536  001021                          BNE     UNSAF           ;BR IF IT WENT UNSAFE
3466  013540  032760  001000  000016          BIT     #BIT09,$TATUS(R0) ;TIMEOUT DURING POSITIONING ?
3467  013546  001022                          BNE     TIMPOS          ;BR IF YES
3468  013550  032760  000400  000016          BIT     #BIT8,$TATUS(R0)  ;TIMEOUT DURING I/O
3469  013556  001023                          BNE     TIMIO           ;BR IF YES
3470  013560  104043                          ERROR   43              ;INVALID STATUS CODE
```

```
                                        END OF ORDER PROCESSING                                                 SEQ 0065

3471  013562  000207                            RTS     PC              ;RETURN
3472
3473                                  ;DRIVE WENT OFFLINE
3474
3475  013564  104021                  OFLIN:    ERROR   21              ;DRIVE WENT OFFLINE
3476  013566  004737  007606                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
3477  013572  000137  011216                    JMP     DROP            ;DEASSIGN THE DRIVE
3478
3479                                  ;PROGRAM TRIED TO GIVE THE DRIVE A BAD COMMAND
3480
3481  013576  104002                  BADOP:    ERROR   2               ;INVALID COMMAND ATTEMPTED
3482  013600  000207                            RTS     PC              ;RETURN
3483
3484                                  ;DRIVE IS UNSAFE
3485
3486  013602  104022                  UNSAF:    ERROR   22              ;DRIVE UNSAFE
3487  013604  004737  007606                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
3488  013610  000137  011216                    JMP     DROP            ;DEASSIGN THE DRIVE
3489
3490                                  ;DRIVE TIMED OUT DURING POSITIONING
3491
3492  013614  104017                  TIMPOS:   ERROR   17              ;DRIVE TIMED OUT DURING POSITIONING
3493  013616  004737  007606                    JSR     PC,INCTOT       ;INCREMENT THE 'TOTAL' ERROR COUNT
3494  013622  000137  011216                    JMP     DROP            ;DEASSIGN THE DRIVE
3495
3496                                  ;DRIVE TIMED OUT DURING A DATA TRANSFER
3497
3498  013626  104020                  TIMIO:    ERROR   20              ;DRIVE TIMED OUT DURING I/O
3499  013630  004737  007606                    JSR     PC,INCTOT       ;INCREMENT THE 'TOTAL' ERROR COUNT
3500  013634  012777  000100  165302            MOV     #BIT06,@$TKS    ;RESTORE THE TTY KEYBOARD INTERRUPT ENABLE
3501  013642  004737  006752                    JSR     PC,CKCLK        ;RESTART THE CLOCK
3502  013646  000137  011216                    JMP     DROP            ;DEASSIGN THE DRIVE
3503
3504                                  ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE' BITS SET
3505
3506  013652  022760  000002  000070  DONE:     CMP     #EOP,$RPER(R0)  ;END OF PACK ?
3507  013660  001475                            BEQ     1$              ;BR IF EOP, OPERATION OVERFLOWED
3508  013662  022760  000010  000070            CMP     #WCE,$RPER(R0)  ;WRITE CHECK ERROR ONLY ?
3509  013670  001002                            BNE     .+6             ;BR IF NOT
3510  013672  000137  014056                    JMP     WCEER           ;REPORT THE WRITE CHECK ERROR
3511  013676  032760  100000  000070            BIT     #WPV,$RPER(R0)  ;WRITE PROTECTION VIOLATION ?
3512  013704  001402                            BEQ     .+6             ;BR IF NOT
3513  013706  000137  014066                    JMP     WPVER           ;REPORT THE WRITE PROTECTION VIOLATION
3514  013712  032760  040000  000070            BIT     #FUV,$RPER(R0)  ;FILE UNSAFE ?
3515  013720  001402                            BEQ     .+6             ;BR IF NOT
3516  013722  000137  013602                    JMP     UNSAF           ;REPORT THE UNSAFE
3517  013726  032760  004000  000066            BIT     #SUSI,$RPDS(R0)  ;SEEK INCOMPLETE ?
3518  013734  001402                            BEQ     .+6             ;BR IF NOT
3519  013736  000137  014414                    JMP     SKIER           ;PROCESS IT
3520  013742  032760  034000  000070            BIT     #NXC!NXT!NXS,$RPER(R0)  ;INVALID DISK ADDRESS ?
3521  013750  001402                            BEQ     .+6             ;BR IF NOT
3522  013752  000137  014076                    JMP     IDADR           ;REPORT INVALID DISK ADDRESS
3523  013756  032760  012000  000070            BIT     #PROG!MODE,$RPER(R0)  ;CONTROLLER ERROR ?
3524  013764  001402                            BEQ     .+6             ;BR IF NOT
3525  013766  000137  014106                    JMP     CNTRLR          ;REPORT THE CONTROLLER ERROR
3526  013772  032760  000024  000070            BIT     #TIMEE NXME,$RPER(R0)  ;INTERFACE (UNIBUS) ERROR ?
```

```
3527  014000  001402                          BEQ     .+6                 ;BR IF NOT
3528  014002  000137  014116                  JMP     INTRFA              ;REPORT THE ERROR
3529  014006  032760  001000  000070          BIT     #FMTE,$RPER(R0)     ;FORMAT ERROR ?
3530  014014  001402                          BEQ     .+6                 ;BR IF NOT
3531  014016  000137  014126                  JMP     FMTERR              ;PROCESS THE FORMAT ERROR
3532  014022  032760  010000  000066          BIT     #HNF,$RPDS(R0)      ;HEADER NOT FOUND ?
3533  014030  001402                          BEQ     .+6                 ;BR IF NOT SET
3534  014032  000137  014450                  JMP     CKHNF               ;PROCESS THE ERROR
3535  014036  032760  000340  000070          BIT     #LPE!WPE!CSME,$RPER(R0)  ;DATA ERROR ?
3536  014044  001402                          BEQ     .+6                 ;BR IF NOT
3537  014046  000137  014612                  JMP     DATER               ;REPORT THE DATA ERROR
3538  014052  104043                          ERROR   43                  ;ERROR REPORT BUT NO BITS SET
3539  014054  000207                  1$:     RTS     PC                  ;RETURN
3540
3541                                  ;REPORT A WRITE CHECK ERROR (DATA CHECK BITS NOT SET)
3542
3543  014056  104032                  WCEER:  ERROR   32                  ;WRITE CHECK (DATA CHECK BITS NOT SET)
3544  014060  004737  007606                  JSR     PC,INCTOT           ;INCREMENT 'TOTAL' ERROR COUNT
3545  014064  000207                          RTS     PC                  ;RETURN
3546
3547                                  ;REPORT WRITE PROTECTION VIOLATION
3548
3549  014066  104044                  WPVER:  ERROR   44                  ;WRITE PROTECTION VIOLATION
3550  014070  004737  007606                  JSR     PC,INCTOT           ;INCREMENT 'TOTAL' ERROR COUNT
3551  014074  000207                          RTS     PC                  ;RETURN
3552
3553                                  ;REPORT INVALID DISK ADDRESS ('NXC', 'NXT', 'NXS')
3554
3555  014076  104026                  BDADR:  ERROR   26                  ;INVALID DISK ADDRESS
3556  014100  004737  007606                  JSR     PC,INCTOT           ;INCREMENT 'TOTAL' ERROR COUNT
3557  014104  000207                          RTS     PC                  ;RETURN
3558
3559                                  ;REPORT CONTROLLER ERROR ('PROG' & 'MODE')
3560
3561  014106  104025                  CNTRLR: ERROR   25                  ;REPORT CONTROLLER ERROR
3562  014110  004737  007606                  JSR     PC,INCTOT           ;INCREMENT 'TOTAL' ERROR COUNT
3563  014114  000207                          RTS     PC                  ;RETURN
3564
3565                                  ;REPORT UNIBUS INTERFACE ERROR ('TIMEE' & 'NXME')
3566
3567  014116  104024                  INTRFA: ERROR   24                  ;REPORT UNIBUS ERROR
3568  014120  004737  007606                  JSR     PC,INCTOT           ;INCREMENT 'TOTAL' ERROR COUNT
3569  014124  000207                          RTS     PC                  ;RETURN
3570
3571                                  ;REPORT FORMAT ERROR
3572
3573  014126  122760  000007  000002  FMTERR: CMPB    #WRTCHK,$COMND(R0)  ;WRITE CHECK OPERATION ?
3574  014134  001404                          BEQ     1$                  ;BR IF WRITE CHECK
3575  014136  132760  000004  000002          BITB    #BIT02,$COMND(R0)   ;WRITE COMMAND ?
3576  014144  001010                          BNE     2$                  ;BR IF NOT
3577  014146  016046  000076          1$:     MOV     $RPBA(R0),-(SP)     ;PUT PRESENT BUFFER ADDRESS ON THE STACK
3578  014152  166016  000006                  SUB     $BUF(R0),(SP)       ;SUBTRACT STARTING ADDRESS
3579  014156  022726  000200                  CMP     #128.,(SP)+         ;SILO FILLED OR HAS XFER TO DISK STARTED ?
3580  014162  103074                          BHIS    6$                  ;BR IF NOT
3581  014164  000404                          BR      3$                  ;
3582  014166  026060  000006  000076  2$:     CMP     $BUF(R0),$RPBA(R0)  ;READ TRANSFER STARTED ?
```

```
3583  014174  001467                          BEQ     6$              ;BR IF NOT
3584  014176  104045                  3$:     ERROR   45              ;FORMAT ERROR DURING I/O
3585  014200  162706  000004                  SUB     #4,SP           ;MAKE ROOM ON THE STACK
3586  014204  004737  006656                  JSR     PC,READDR       ;ADJUST SECTOR/TRACK ADDRESSES
3587  014210  112660  000013                  MOVB    (SP)+,$TRK(R0)  ;MOVE THE TRACK ADDRESS
3588  014214  112660  000012                  MOVB    (SP)+,$SEC(R0)  ;MOVE THE SECTOR ADDRESS
3589  014220  016060  000106  000010          MOV     $SUCA(R0),$CYL(R0)  ;USE PRESENT CYLINDER ADDRESS
3590  014226  022760  000400  000020          CMP     #256.,$WRDL(R0) ;WAS INITIAL TRANSFER TO BE GREATER
3591                                                                  ;THAN 1 SECTOR ?
3592  014234  103015                          BHIS    4$              ;BR IF NOT
3593  014236  004737  005046                  JSR     PC,RELBUF       ;RELEASE PRESENT BUFFER
3594  014242  012760  000400  000020          MOV     #256.,$WRDL(R0) ;SETUP FOR THE FAILING SECTOR
3595  014250  012760  177400  000004          MOV     #-256.,$WRDM(R0)  ;WORD COUNT (2'S COMPLEMENT)
3596  014256  005046                          CLR     -(SP)           ;CLEAR THE STACK
3597  014260  004737  004710                  JSR     PC,GETBUF       ;GET NEW BUFFER
3598  014264  012660  000006                  MOV     (SP)+,$BUF(R0)  ;LOAD BUFFER ADDRESS
3599  014270  012737  001001  015562  4$:     MOV     #FMTE!DSKERR,MASK       ;ERROR MASK FOR RETRY
3600  014276  112737  000012  015561          MOVB    #10.,$RETRY+1   ;RETRY LIMIT
3601  014304  004737  007606                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3602  014310  004737  015462                  JSR     PC,RETRY        ;RETRY THE OPERATION
3603  014314  000411                          BR      5$              ;RETRY GOOD
3604  014316  004737  015340                  JSR     PC,RECALT       ;RESTORE (PART OF STANDARD RECOVERY)
3605  014322  004737  015462                  JSR     PC,RETRY        ;RETRY AGAIN
3606  014326  000404                          BR      5$              ;RETRY GOOD
3607  014330  104042                          ERROR   42              ;UNSUCESSFUL RETRY
3608  014332  004737  007534                  JSR     PC,INCHRD       ;INCREMENT 'HARD' ERROR COUNT
3609  014336  000425                          BR      8$              ;EXIT
3610  014340  104041                  5$:     ERROR   41              ;SUCESSFUL RETRY
3611  014342  004737  007516                  JSR     PC,INCSOF       ;INCREMENT 'SOFT' ERROR COUNT
3612  014346  004737  012524                  JSR     PC,CMPAR        ;COMPARE THE BUFFER
3613  014352  000417                          BR      8$              ;EXIT
3614  014354  104035                  6$:     ERROR   35              ;FORMAT ERROR SEARCHING FOR INITIAL SECTOR
3615  014356  004737  007606                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3616  014362  012737  001001  015562          MOV     #FMTE!DSKERR,MASK   ;ERROR MASK FOR RETRY
3617  014370  112737  000001  015561          MOVB    #1,$RETRY+1     ;RETRY COUNT
3618  014376  004737  015462                  JSR     PC,RETRY        ;RETRY THE ORDER
3619  014402  000402                          BR      7$              ;GOOD RETRY
3620  014404  104042                          ERROR   42              ;UNSUCESSFUL RETRY
3621  014406  000401                          BR      8$              ;EXIT
3622  014410  104041                  7$:     ERROR   41              ;SUCESSFUL RETRY
3623  014412  000207                  8$:     RTS     PC              ;RETURN
3624
3625                                  ;REPORT SEEK INCOMPLETE
3626
3627  014414  016037  000032  001124  SKIER:  MOV     $PREVA+2(R0),$GDDAT     ;PREVIOUS CYLINDER ADDRESS
3628  014422  016037  000010  001126          MOV     $CYL(R0),$BDDAT ;DESTINATION CYLINDER
3629  014430  104023                          ERROR   23              ;SEEK INCOMPLETE
3630  014432  004737  007606                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3631  014436  004737  007552                  JSR     PC,INCSKI       ;INCREMENT 'SUSI' ERROR COUNT
3632  014442  004737  015340                  JSR     PC,RECALT       ;RESTORE
3633  014446  000207                          RTS     PC              ;RETURN
3634
3635                                  ;REPORT HEADER NOT FOUND ERROR OR POSITIONING ERROR
3636
3637  014450  116046  000102          CKHNF:  MOVB    $SDDA(R0),-(SP) ;SECTOR ADDRESS ON THE STACK
3638  014454  042716  000360                  BIC     #360,(SP)       ;CLEAR THE 'SOT' BITS
```

```
3639  014460  116046  000103                    MOVB    $RPDA+1(R0),-(SP)  ;TRACK ADDRESS
3640  014464  004737  015372                    JSR     PC,READHD          ;READ THE HEADER FROM THE INITIAL SECTOR
3641  014470  032737  100000  025426            BIT     #ERR,GENREG+RPER   ;ERROR DURING OPERATION ?
3642  014476  001041                            BNE     1$                 ;BR IF ERROR
3643  014500  013737  033630  001126            MOV     CYLDER+2,$BDDAT    ;CYLINDER FROM HEADER
3644  014506  006237  001126                    ASR     $BDDAT             ;SHIFT CYLINDER ADDRESS
3645  014512  006237  001126                    ASR     $BDDAT             ;SHIFT CYLINDER ADDRESS
3646  014516  006237  001126                    ASR     $BDDAT             ;SHIFT CYLINDER ADDRESS
3647  014522  006237  001126                    ASR     $BDDAT             ;SHIFT CYLINDER ADDRESS
3648  014526  006237  001126                    ASR     $BDDAT             ;SHIFT CYLINDER ADDRESS
3649  014532  006237  001126                    ASR     $BDDAT             ;SHIFT CYLINDER ADDRESS
3650  014536  026037  000106  001126            CMP     $SUCA(R0),$BDDAT   ;AT CORRECT CYLINDER ?
3651  014544  001416                            BEQ     1$                 ;BR IF AT CORRECT CYLINDER
3652  014546  016037  000032  001124            MOV     $PREVA+2(R0),$GDDAT  ;PREVIOUS CYLINDER ADDRESS
3653  014554  104036                            ERROR   36                 ;POSITIONING ERROR
3654  014556  004737  007570                    JSR     PC,INCMIS          ;INCREMENT POSITIONING ERROR COUNT
3655  014562  004737  015340                    JSR     PC,RECALT          ;RESTORE THE DRIVE
3656  014566  016037  000106  000032            MOV     $SUCA(R0),$PREVA+2  ;SAVE PREVIOUS CYLINDER
3657  014574  005060  000010                    CLR     $CYL(R0)           ;RESET CURRENT CYLINDER
3658  014600  000401                            BR      2$                 ;CLEAN UP, THEN EXIT
3659  014602  104034                    1$:     ERROR   34                 ;HEADER NOT FOUND ERROR
3660  014604  004737  007606            2$:     JSR     PC,INCTOT          ;INCREMENT 'TOTAL' ERROR COUNT
3661  014610  000207                            RTS     PC                 ;RETURN
3662
3663                                    ;REPORT DATA ERROR ('LPE', 'WPE', & 'CSME')
3664
3665  014612  032760  000010  000070    DATER:  BIT     #WCE,$RPER(R0)     ;WRITE CHECK ERROR ALSO ?
3666  014620  001402                            BEQ     1$                 ;BR IF NOT
3667  014622  104033                            ERROR   33                 ;REPORT WRITE CHECK WITH DATA CHECK
3668  014624  000403                            BR      2$                 ;CONTINUE
3669  014626  104027                    1$:     ERROR   27                 ;DATA ERROR
3670  014630  004737  015134                    JSR     PC,PRTBAD          ;TYPE THE BAD SECTOR
3671  014634  004737  007606            2$:     JSR     PC,INCTOT          ;INCREMENT 'TOTAL' ERROR COUNT
3672  014640  162706  000004                    SUB     #4,SP              ;MAKE ROOM ON THE STACK
3673  014644  004737  006656                    JSR     PC,READDR          ;ADJUST SECTOR/TRACK ADDRESSES
3674  014650  112660  000013                    MOVB    (SP)+,$TRK(R0)     ;MOVE THE TRACK ADDRESS
3675  014654  112660  000012                    MOVB    (SP)+,$SEC(R0)     ;MOVE THE SECTOR ADDRESS
3676  014660  016060  000106  000010            MOV     $SUCA(R0),$CYL(R0)  ;USE PRESENT CYLINDER ADDRESS
3677  014666  022760  000400  000020            CMP     #256.,$WRDL(R0)    ;INITIAL TRANSFER SIZE LARGER
3678                                                                       ;THAN 1 SECTOR ?
3679  014674  103023                            BHIS    3$                 ;BR IF NOT
3680  014676  004737  005046                    JSR     PC,RELBUF          ;RELEASE THE PRESENT BUFFER
3681  014702  012760  000400  000020            MOV     #256.,$WRDL(R0)    ;SETUP FOR 1 SECTOR
3682  014710  012760  177400  000004            MOV     #-256.,$WRDM(R0)   ;WORD COUNT
3683  014716  005046                            CLR     -(SP)              ;CLEAR THE STACK
3684  014720  004737  004710                    JSR     PC,GETBUF          ;GET NEW BUFFER
3685  014724  012660  000006                    MOV     (SP)+,$BUF(R0)     ;LOAD BUFFER ADDRESS
3686  014730  122760  000007  000002            CMPB    #WRTCHK,$COMND(R0)  ;IS THE COMMAND A WRITE CHECK ?
3687  014736  001002                            BNE     3$                 ;BR IF NOT
3688  014740  004737  005274                    JSR     PC,FILBUF          ;REFILL THE BUFFER
3689  014744  012737  000350  015562    3$:     MOV     #WCE!LPE!WPE,CSME,MASK  ;RETRY MASK
3690  014752  112737  000012  015561            MOVB    #10.,$RETRY+1      ;RETRY LIMIT
3691  014760  004737  015462                    JSR     PC,RETRY           ;RETRY THE SECTOR
3692  014764  000413                            BR      4$                 ;RETRY GOOD
3693  014766  004737  015340                    JSR     PC,RECALT          ;RESTORE THE DRIVE
3694  014772  004737  015462                    JSR     PC,RETRY           ;RETRY 10 MORE TIMES
```

```
3695  014776  000406                          BR      4$              ;RETRY GOOD
3696  015000  104031                          ERROR   31              ;HARD ERROR
3697  015002  004737  007534                  JSR     PC,INCHRD       ;INCREMENT 'HARD' ERROR COUNT
3698  015006  004737  015134                  JSR     PC,PRTBAD       ;PRINT BAD SECTOR ?
3699  015012  000405                          BR      5$              ;EXIT
3700  015014  104030                  4$:     ERROR   30              ;SOFT ERROR
3701  015016  004737  007516                  JSR     PC,INCSOF       ;INCREMENT 'SOFT' ERROR COUNT
3702  015022  004737  012524                  JSR     PC,CMPAR        ;COMPARE THE BUFFER
3703  015026  000207                  5$:     RTS     PC              ;RETURN
3704
3705
3706                                  ;REPORT AN 'UNKNOWN' DATA PATTERN
3707
3708  015030  010346                  NOMTCH: MOV     R3,-(SP)        ;SAVE R3
3709  015032  112737  000010  001114          MOVB    #10,$ITEMB      ;ERROR TABLE INDEX
3710  015040  004737  012120                  JSR     PC,TYPERR       ;CAN'T MATCH PATTERN WITH DATA READ
3711  015044  012703  000004                  MOV     #4,R3           ;WORD COUNTER
3712  015050  010137  001122          1$:     MOV     R1,$BDADR       ;ADDRESS OF WORD
3713  015054  012137  001126                  MOV     (R1)+,$BDDAT    ;WORD
3714  015060  112737  000011  001114          MOVB    #11,$ITEMB      ;ERROR TABLE INDEX
3715  015066  004737  012120                  JSR     PC,TYPERR       ;DISPLAY THE PATTERN 'SEED'
3716  015072  005303                          DEC     R3              ;DECREMENT WORD COUNTER
3717  015074  001365                          BNE     1$              ;BR IF NOT DONE
3718  015076  062701  000770                  ADD     #770,R1         ;INCREMENT BUFFER POINTER
3719  015102  060237  013140                  ADD     R2,ERCTR        ;ADD SECTOR (OR LESS) COUNT TO THE
3720                                                                  ;COMPARSION ERROR COUNT
3721  015106  005002                          CLR     R2              ;CLEAR 'WORDS TO COMPARE' COUNT IN R2
3722  015110  112737  177777  013133          MOVB    #-1,FRSTER      ;SET 'NOT FIRST ERROR' INDICATOR
3723  015116  012603                          MOV     (SP)+,R3        ;RESTORE R3
3724  015120  032777  100000  164012          BIT     #SW15,@SWR      ;HALT ON ERROR ?
3725  015126  001401                          BEQ     2$              ;BR IF NOT
3726  015130  000000                          HALT                    ;ERROR HALT
3727  015132  000207                  2$:     RTS     PC              ;RETURN
3728
3729                                  ;ROUTINE TO DISPLAY THE SECTOR WHICH GAVE THE HARD ERROR
3730
3731  015134  032777  000010  163776  PRTBAD: BIT     #SW3,@SWR       ;PRINT THE BAD SECTOR ?
3732  015142  001475                          BEQ     7$              ;BR IF NOT
3733  015144  032777  020000  163766          BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS
3734  015152  001071                          BNE     7$              ;BR IF INHIBIT
3735  015154  122760  000007  000002          CMPB    #WRTCHK,$COMND(R0)  ;PRESENT COMMAND A WRITE CHECK ?
3736  015162  001465                          BEQ     7$              ;BR IF IT IS
3737  015164  032760  000004  000002          BIT     #BIT02,$COMND(R0)   ;PRESENT COMMAND A WRITE ?
3738  015172  001461                          BEQ     7$              ;BR IF IT IS
3739  015174  016001  000076                  MOV     $RPBA(R0),R1    ;PUT THE END ADDRESS INTO R1
3740  015200  016046  000020                  MOV     $WRDL(R0),-(SP) ;FIND THE BEGINNING OF THE SECTOR
3741  015204  066016  000074                  ADD     $RPWC(R0),(SP)  ;SUBTRACT THE WORDS NOT TRANSFERED
3742  015210  005046                          CLR     -(SP)           ;MAKE THE UPPER DIVIDEND 0
3743  015212  012746  000400                  MOV     #256.,-(SP)  ;DIVIDE THE WORDS TRANSFERED BY THE SECTOR SIZE
3744  015216  004737  020176                  JSR     PC,$DIV         ;DIVIDE
3745  015222  005716                          TST     (SP)            ;REMANDER = 0 ?
3746  015224  001403                          BEQ     1$              ;BR IF IT IS - COMPLETE SECTOR TRANSFERED
3747  015226  006316                          ASL     (SP)            ;CONVERT THE RESIDUAL SECTOR SIZE INTO BYTE COUNT
3748  015230  161601                          SUB     (SP),R1         ;SUBTRACT IT FROM THE END ADDRESS
3749  015232  000402                          BR      2$              ;FINISH THE SIZING
3750  015234  162701  001000          1$:     SUB     #1000,R1        ;SUBTRACT FULL SECTOR SIZE FROM END ADDR
```

```
3751  015240  062706  000004          2$:     ADD     #4,SP           ;RESTORE THE STACK POINTER
3752  015244  104401  032374                  TYPE    ,LIN11H         ;PRINT THE HEADER
3753  015250  032777  000010  163662  3$:     BIT     #SW03,@SWR      ;PRINT ANY MORE OF THE SECTOR
3754  015256  001420                          BEQ     5$              ;BR IF NOT
3755  015260  012702  000007                  MOV     #7,R2           ;R2 CONTAINS THE WORDS/LINE COUNT
3756  015264  010146                          MOV     R1,-(SP)        ;PUT THE ADDRESS ON THE STACK
3757  015266  104402                          TYPOC                   ;PRINT IT
3758  015270  020160  000076          4$:     CMP     R1,$RPBA(R0)    ;PRINTED ALL THE SECTOR ?
3759  015274  001411                          BEQ     5$              ;BR IF ALL PRINTED
3760  015276  104401  032470                  TYPE    ,SPACE2         ;SPACES
3761  015302  012146                          MOV     (R1)+,-(SP)     ;PUT THE DATA ON THE STACK
3762  015304  104402                          TYPOC                   ;PRINT IT
3763  015306  005302                          DEC     R2              ;DECREMENT THE HORIZONTAL COUNT
3764  015310  001367                          BNE     4$              ;BR IF NOT AT THE END OF THE LINE
3765  015312  104401  001213                  TYPE    ,$CRLF          ;CR-LF
3766  015316  000754                          BR      3$              ;RESTORE THE WORDS/LINE COUNT
3767  015320  104401  001213          5$:     TYPE    ,$CRLF          ;PRINT WHAT REMAINS IN THE BUFFER
3768  015324  032777  100000  163606  6$:     BIT     #SW15,@SWR      ;HALT ON ERROR ?
3769  015332  001401                          BEQ     7$              ;BR IF NOT
3770  015334  000000                          HALT                    ;HALT
3771  015336  000207                  7$:     RTS     PC              ;RETURN
3772
3773                                  ;ROUTINE TO DO A HOME SEEK - DRIVE SELECTED IN R0
3774                                          ;ENTER AT 'RECAL0' IF UNIT NUMBER ALREADY LOADED
3775                                          ;INTO 'GENDPB'
3776
3777  015340  111037  025404          RECALT: MOVB    (R0),GENDPB     ;MOVE THE UNIT # TO THE GENERAL DPB
3778  015344  112737  000015  025406  RECAL0: MOVB    #HOMSEK,GENDPB+$COMND  ;HOME SEEK COMMAND
3779  015352  004037  021556                  JSR     R0,RP11         ;DRIVER ENTRANCE
3780  015356  025404                          GENDPB                  ;DPB ADDRESS FOR ORDER
3781  015360  000000                          HALT                    ;DRIVER DIDN'T ACCEPT THE ORDER
3782  015362  005737  025422          1$:     TST     GENDPB+$TATUS           ;SEE IF FINISHED
3783  015366  001775                          BEQ     1$              ;BR IF NOT FINISHED
3784  015370  000207                          RTS     PC              ;RETURN
3785
3786                                  ;UTILITY READ HEADER ROUTINE
3787                                  ;       ENTER WITH TRACK AND SECTOR ADDRS ON THE STACK
3788
3789  015372  116637  000002  025417  READHD: MOVB    2(SP),GENDPB+$TRK ;TRACK ADDRESS
3790  015400  116637  000004  025416          MOVB    4(SP),GENDPB+$SEC ;SECTOR ADDRESS
3791  015406  111037  025404                  MOVB    (R0),GENDPB     ;DRIVE NUMBER
3792  015412  016037  000106  025414          MOV     $SUCA(R0),GENDPB+$CYL ;CYLINDER ADDRESS
3793  015420  012737  014000  025406          MOV     #MODE!HDR,GENDPB+$COMND  ;MODE AND HEADER BITS
3794  015426  112737  000017  025406          MOVB    #READ,GENDPB+$COMND ;READ COMMAND
3795  015434  004037  021556                  JSR     R0,RP11         ;DRIVER ENTRANCE
3796  015440  025404                          GENDPB                  ;DPB ADDRESS FOR ORDER
3797  015442  000000                          HALT                    ;DRIVER DIDN'T ACCEPT COMMAND
3798  015444  005737  025422          1$:     TST     GENDPB+$TATUS   ;FINISHED?
3799  015450  001775                          BEQ     1$              ;BR IF NOT
3800  015452  012666  000002                  MOV     (SP)+,2(SP)     ;ADJUST STACK FOR RETURN
3801  015456  005726                          TST     (SP)+
3802  015460  000207                          RTS     PC              ;RETURN
3803
3804                                  ;RETRY THE PRESENT OPERATION
3805                                  ;       RETURN IF RETRY SUCESSFUL
3806                                  ;       RETURN+2 IF RETRY UNSUCESSFUL
```

END OF ORDER PROCESSING SEQ 0071

```
3807                                    ;       RETURN TO MAIN PROGRAM IF DIFFERENT ERROR OCCURS
3808
3809  015462  105037  015560            RETRY:  CLRB    $RETRY          ;CLEAR RETRY COUNTER
3810  015466  004737  005424            1$:     JSR     PC,GODRIV       ;RE-START ORDER
3811  015472  005760  000016            2$:     TST     $STATUS(R0)     ;ORDER FINISHED?
3812  015476  001775                            BEQ     2$              ;BR IF NOT
3813  015500  100403                            BMI     3$              ;BR IF ERROR
3814  015502  105237  015560                    INCB    $RETRY          ;INCREMENT RETRY COUNT
3815  015506  000420                            BR      6$              ;GO TO EXIT, SUCESSFUL RETRY
3816  015510  032760  000200  000016  3$:     BIT     #BIT7,$STATUS(R0)  ;DID ORDER TERMINATE NORMALLY ?
3817  015516  001415                            BEQ     7$              ;BR IF NOT
3818  015520  033760  015562  000070            BIT     MASK,$RPER(R0)   ;SAME ERROR?
3819  015526  001411                            BEQ     7$              ;BR IF NOT
3820  015530  105237  015560            5$:     INCB    $RETRY          ;INCREMENT RETRY COUNT
3821  015534  123737  015560  015561            CMPB    $RETRY,$RETRY+1 ;REACHED RETRY LIMIT ?
3822  015542  001351                            BNE     1$              ;BR IF NOT
3823  015544  062716  000002                    ADD     #2,(SP)         ;INCREMENT RETURN, UNSUCESSFUL RETRY
3824  015550  000207                    6$:     RTS     PC              ;RETURN
3825  015552  104037                    7$:     ERROR   37              ;DIFFERENT ERROR DURING RETRY
3826  015554  005726                            TST     (SP)+           ;ADJUST STACK POINTER FOR DIRECT RETURN
3827  015556  000207                            RTS     PC              ;RETURN
3828
3829  015560     000     000            $RETRY: .BYTE   0,0             ;RETRY LIMIT & RETRY COUNTER
3830  015562  000000                    MASK:   .WORD   0               ;ERROR MASK
3831
3832                                    ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO
3833                                    ;AN UNSIGNED DECIMAL ASCIZ NUMBER.
3834                                    ;
3835                                    ;CALL
3836                                    ;       MOV     #NUMBER,-(SP)   ;PUT THE BINARY NUMBER ON THE STACK
3837                                    ;       JSR     PC,$SB2D        ;CALL
3838                                    ;       RETURN                  ;ADDRESS OF 1ST CHARACTER IS ON THE STACK
3839                                    ;
3840                                    ;NOTE:  THIS FORM OF '$SB2D' MUST BE USED, THE PROGRAM CANNOT USE THE
3841                                    ;VERSION OF THIS ROUTINE WHICH IS ON THE SYSMAC LIBRARY, REV C OR LATER.
3842
3843  015564  016637  000002  015610  $SB2D:  MOV     2(SP),1$        ;SAVE THE BINARY NUMBER
3844  015572  012746  015610                    MOV     #1$,-(SP)       ;SET POINTER
3845  015576  004737  020614                    JSR     PC,$DB2D        ;CALL DOUBLE LENGTH CONVERT
3846  015602  012666  000002                    MOV     (SP)+,2(SP)     ;PICKUP THE POINTER
3847  015606  000207                            RTS     PC              ;RETURN
3848  015610  000000  000000            1$:     .WORD   0,0
3849
3850                                    ;;*****************************************************************
3851
3852                                    .SBTTL  MACRO ROUTINES
3853
3854                                    .SBTTL  ERROR HANDLER ROUTINE
3855
3856                                    ;;*****************************************************************
3857                                    ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3858                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3859                                    ;*AND GO TO TYPERR ON ERROR
3860                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3861                                    ;*SW15=1         HALT ON ERROR
3862                                    ;*SW13=1         INHIBIT ERROR TYPEOUTS
```

ERROR HANDLER ROUTINE SEQ 0072

```
3863                                    ;*SW10=1        BELL ON ERROR
3864                                    ;*CALL
3865                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3866
3867  015614                            $ERROR:
3868  015614  104407                            CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
3869  015616  010146                            MOV     R1,-(SP)        ;SAVE R1
3870  015620  010246                            MOV     R2,-(SP)        ;SAVE R2
3871  015622  111037  001264                    MOVB    (R0),DRIVE      ;DRIVE NUMBER
3872  015626  162706  000004                    SUB     #4,SP           ;MAKE ROOM ON THE STACK
3873  015632  004737  006656                    JSR     PC,READDR       ;DECREMENT THE TRACK/SECTOR ADDRESSES
3874  015636  112637  001270                    MOVB    (SP)+,TRK.ER    ;TRACK ADDRESS
3875  015642  112637  001272                    MOVB    (SP)+,SEC.ER    ;SECTOR ADDRESS
3876  015646  016037  000106  001266            MOV     $SUCA(R0),CYL.ER  ;CYLINDER ADDRESS
3877  015654  012701  001162                    MOV     #$REG0,R1       ;START AT '$REG0'
3878  015660  016021  000066                    MOV     $RPDS(R0),(R1)+ ;MOVE RPDS
3879  015664  016021  000070                    MOV     $RPER(R0),(R1)+ ;MOVE RPER
3880  015670  016021  000072                    MOV     $RPCS(R0),(R1)+ ;MOVE RPCS
3881  015674  016021  000074                    MOV     $RPWC(R0),(R1)+ ;MOVE RPWC
3882  015700  016021  000076                    MOV     $RPBA(R0),(R1)+ ;MOVE RPBA
3883  015704  016021  000100                    MOV     $RPCA(R0),(R1)+ ;MOVE RPCA
3884  015710  016021  000102                    MOV     $RPDA(R0),(R1)+ ;MOVE RPDA
3885  015714  016021  000104                    MOV     $RPM1(R0),(R1)+ ;MOVE RPM1
3886  015720  016021  000106                    MOV     $SUCA(R0),(R1)+ ;MOVE SUCA
3887  015724  016021  000110                    MOV     $SILO(R0),(R1)+ ;MOVE SILO
3888  015730  012602                            MOV     (SP)+,R2        ;RESTORE R2
3889  015732  012601                            MOV     (SP)+,R1        ;RESTORE R1
3890  015734  105237  001103           7$:      INCB    $ERFLG          ;;SET THE ERROR FLAG
3891  015740  001775                            BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
3892  015742  013777  001102  163172            MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
3893  015750  032777  002000  163162            BIT     #BIT10,@SWR     ;;BELL ON ERROR?
3894  015756  001402                            BEQ     1$              ;;NO - SKIP
3895  015760  104401  001206                    TYPE    ,$BELL          ;;RING BELL
3896  015764  005237  001112           1$:      INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
3897  015770  011637  001116                    MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
3898  015774  162737  000002  001116            SUB     #2,$ERRPC
3899  016002  117737  163110  001114            MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
3900  016010  032777  020000  163122            BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
3901  016016  001004                            BNE     20$             ;;SKIP TYPEOUTS
3902  016020  004737  012120                    JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
3903  016024  104401  001213                    TYPE    ,$CRLF
3904  016030                            20$:
3905  016030  005777  163104           2$:      TST     @SWR            ;;HALT ON ERROR
3906  016034  100002                            BPL     3$              ;;SKIP IF CONTINUE
3907  016036  000000                            HALT                    ;;HALT ON ERROR!
3908  016040  104407                            CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
3909  016042                            3$:
3910  016042  000002                            RTI                     ;;RETURN
3911
3912                                    .SBTTL  TYPE ROUTINE
3913
3914                                    ;;**********************************************************************
3915                                    ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3916                                    ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3917                                    ;*NOTE1:        $FILL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3918                                    ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
```

```
3919                                    ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3920                                    ;*
3921                                    ;*CALL:
3922                                    ;*1) USING A TRAP INSTRUCTION
3923                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3924                                    ;*OR
3925                                    ;*      TYPE
3926                                    ;*      MESADR
3927                                    ;*
3928
3929  016044  105737  001157           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
3930  016050  100002                          BPL     1$              ;;BR IF YES
3931  016052  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
3932  016054  000407                          BR      3$              ;;LEAVE
3933  016056  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
3934  016060  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
3935  016064  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3936  016066  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
3937  016070  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
3938  016072  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
3939  016074  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
3940  016100  000002                          RTI                     ;;RETURN
3941  016102  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
3942  016106  001430                          BEQ     8$
3943  016110  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
3944  016114  001006                          BNE     5$
3945  016116  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
3946  016120  104401                          TYPE                    ;;TYPE A CR AND LF
3947  016122  001213                          $CRLF
3948  016124  105037  016260                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
3949  016130  000755                          BR      2$              ;;GET NEXT CHARACTER
3950  016132  004737  016214          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
3951  016136  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3952  016142  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
3953  016144  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3954                                                                  ;;AND THE NULL CHAR.
3955  016150  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3956  016154  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3957  016156  004737  016214                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
3958  016162  105337  016260                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
3959  016166  000770                          BR      7$              ;;LOOP
3960
3961                                    ;HORIZONTAL TAB PROCESSOR
3962
3963  016170  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3964  016174  004737  016214          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3965  016200  132737  000007  016260          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3966  016206  001372                          BNE     9$              ;;TAB STOP
3967  016210  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
3968  016212  000724                          BR      2$              ;;GET NEXT CHARACTER
3969  016214  105777  162730          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3970  016220  100375                          BPL     $TYPEC
3971  016222  116677  000002  162722          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3972  016230  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3973  016236  001003                          BNE     1$              ;;BRANCH IF NO
3974  016240  105037  016260                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
```

```
 3975  016244  000406                          BR      $TYPEX          ;;EXIT
 3976  016246  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 3977  016254  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 3978  016256  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 3979  016260  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 3980  016262  000207                  $TYPEX: RTS     PC
 3981
 3982
 3983                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 3984
 3985                                  ;;**************************************************************************
 3986                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 3987                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 3988                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 3989                                  ;*CALL:
 3990                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3991                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
 3992                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 3993                                  ;*      .BYTE   M               ;;M=1 OR 0
 3994                                  ;*                                      ;;1=TYPE LEADING ZEROS
 3995                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
 3996                                  ;*
 3997                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 3998                                  ;*$TYPOS OR $TYPOC
 3999                                  ;*CALL:
 4000                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 4001                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
 4002                                  ;*
 4003                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 4004                                  ;*CALL:
 4005                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 4006                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
 4007
 4008  016264  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 4009  016270  116637  000001  016507          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 4010  016276  112637  016511                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 4011  016302  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 4012  016306  000406                          BR      $TYPON
 4013  016310  112737  000001  016507  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 4014  016316  112737  000006  016511          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 4015  016324  112737  000005  016506  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 4016  016332  010346                          MOV     R3,-(SP)        ;;SAVE R3
 4017  016334  010446                          MOV     R4,-(SP)        ;;SAVE R4
 4018  016336  010546                          MOV     R5,-(SP)        ;;SAVE R5
 4019  016340  113704  016511                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 4020  016344  005404                          NEG     R4
 4021  016346  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 4022  016352  110437  016510                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 4023  016356  113704  016507                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 4024  016362  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 4025  016366  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
 4026  016370  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 4027  016372  000404                          BR      3$              ;;GO DO MSB
 4028  016374  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
 4029  016376  006105                          ROL     R5
 4030  016400  006105                          ROL     R5
```

```
                                        MOV     R5,R3
                                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
016406 105337 016510                    DECB    $OMODE          ;;TYPE THIS DIGIT?
016412 100016                           BPL     7$              ;;BR IF NO
016414 042703 177770                    BIC     #177770,R3      ;;GET RID OF JUNK
016420 001002                           BNE     4$              ;;TEST FOR 0
016422 005704                           TST     R4              ;;SUPPRESS THIS 0?
016424 001403                           BEQ     5$              ;;BR IF YES
016426 005204                   4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
016430 052703 000060                    BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
016434 052703 000040            5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
016440 110337 016504                    MOVB    R3,8$           ;;SAVE FOR TYPING
016444 104401 016504                    TYPE    ,8$             ;;GO TYPE THIS DIGIT
016450 105337 016506            7$:     DECB    $OCNT           ;;COUNT BY 1
016454 003347                           BGT     2$              ;;BR IF MORE TO DO
016456 002402                           BLT     6$              ;;BR IF DONE
016460 005204                           INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
016462 000744                           BR      2$              ;;GO DO THE LAST DIGIT
016464 012605                   6$:     MOV     (SP)+,R5        ;;RESTORE R5
016466 012604                           MOV     (SP)+,R4        ;;RESTORE R4
016470 012603                           MOV     (SP)+,R3        ;;RESTORE R3
016472 016666 000002 000004             MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
016500 012616                           MOV     (SP)+,(SP)
016502 000002                           RTI                     ;;RETURN
016504    000                   8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
016505    000                           .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
016506    000                   $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
016507    000                   $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
016510 000000                   $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE

                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

016512                          $TYPDS:
016512 010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016514 010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016516 010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016520 010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016522 010546                           MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016524 012746 020200                    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
016530 016605 000020                    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
016534 100004                           BPL     1$              ;;BR IF INPUT IS POS.
016536 005405                           NEG     R5              ;;MAKE THE BINARY NUMBER POS.
016540 112766 000055 000001             MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
016546 005000                   1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
016550 012703 016726                    MOV     #2$BLK,R3       ;;SETUP THE OUTPUT POINTER
016554 112723 000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
```

```
 4087  016560  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 4088  016562  016001  016716                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 4089  016566  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 4090  016570  002402                          BLT     4$              ;;BR IF DONE
 4091  016572  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 4092  016574  000774                          BR      3$
 4093  016576  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 4094  016600  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
 4095  016602  001002                          BNE     5$              ;;FALL THROUGH IF 0
 4096  016604  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 4097  016606  100407                          BMI     7$              ;;BR IF YES
 4098  016610  106316                  5$:     ASLB    (SP)            ;;MSD?
 4099  016612  103003                          BCC     6$              ;;BR IF NO
 4100  016614  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 4101  016622  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 4102  016626  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 4103  016632  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 4104  016634  005720                          TST     (R0)+           ;;JUST INCREMENTING
 4105  016636  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
 4106  016642  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
 4107  016644  003002                          BGT     8$              ;;GO TO EXIT
 4108  016646  010502                          MOV     R5,R2           ;;GET THE LSD
 4109  016650  000764                          BR      6$              ;;GO CHANGE TO ASCII
 4110  016652  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 4111  016654  100003                          BPL     9$              ;;BR IF NO
 4112  016656  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 4113  016664  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 4114  016666  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 4115  016670  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 4116  016672  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 4117  016674  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 4118  016676  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 4119  016700  104401  016726                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4120  016704  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4121  016712  012616                          MOV     (SP)+,(SP)
 4122  016714  000002                          RTI                     ;;RETURN TO USER
 4123  016716  023420                  $DTBL:  10000.
 4124  016720  001750                          1000.
 4125  016722  000144                          100.
 4126  016724  000012                          10.
 4127  016726  000004                  $DBLK:  .BLKW   4
 4128
 4129                                  .SBTTL  TTY INPUT ROUTINE
 4130
 4131                                  ;;**************************************************************************
 4132                                  .ENABL  LSB
 4133  016736  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
 4134  016740  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
 4135  016742  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
 4136  016744  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
 4137          016745                  $TKQEND .
 4138          016746                  .EVEN
 4139
 4140                                  ;*TK INITIALIZE ROUTINE
 4141                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 4142                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
```

```
                                        ;
                                        ;*CALL:
                                        ;*      JSR     PC,$TKINT
                                        ;*      RETURN
                                        ;
016746  005037  016736          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
016752  012737  016744  016740          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
016760  013737  016740  016742          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
016766  012737  017016  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
016774  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;'BR' LEVEL 4
017002  005777  162140                  TST     @$TKB           ;;CLEAR DONE FLAG
017006  012777  000100  162130          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
017014  000207                          RTS     PC              ;;RETURN TO CALLER
                                        ;
                                        ;*TK SERVICE ROUTINE
                                        ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                        ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                        ;*IT IN THE QUEUE.
                                        ;*IF THE CHARACTER IS A ''CONTROL-C'' (^C) $TKINT IS CALLED AND
                                        ;*UPON RETURN EXIT IS MADE TO THE ''CONTROL-C'' RESTART ADDRESS (@CNTRLC)
                                        ;
017016  117746  162124          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
017022  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
017026  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
017032  001007                          BNE     1$              ;;BRANCH IF NO
017034  104401  020134                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
017040  004737  016746                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
017044  005726                          TST     (SP)+           ;;CLEAN UP STACK
017046  000177  162170                  JMP     @CNTRLC         ;;CONTROL C RESTART
017052  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
017056  001004                          BNE     2$              ;;BRANCH IF NO
017060  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
017066  001500                          BEQ     6$              ;;GO TO SWR CHANGE
                                        ;
017070                          2$:
017070  022737  000001  016736          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
017076  001004                          BNE     3$              ;;BRANCH IF NO
017100  104401  001206                  TYPE    ,$BELL          ;;RING THE TTY BELL
017104  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
017106  000451                          BR      5$              ;;EXIT
017110  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
017114  001021                          BNE     32$             ;;BRANCH IF NO
017116  005077  162022                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
017122  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
017124  105777  162014          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
017130  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
017132  117746  162010                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
017136  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
017142  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
017146  001366                          BNE     31$             ;;BRANCH IF NO
017150  012777  000100  161766          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
017156  000002                          RTI                     ;;RETURN
017160  005237  016736          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
017164  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
017170  002405                          BLT     4$              ;;BRANCH IF YES
017172  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
```

```
 4199  017176  003002                          BGT     4$                      ;;BRANCH IF YES
 4200  017200  042716  000040                  BIC     #40,(SP)                ;;MAKE IT UPPER CASE
 4201  017204  112677  177530          4$:     MOVB    (SP)+,@$TKQIN           ;;AND PUT IT IN QUEUE
 4202  017210  005237  016740                  INC     $TKQIN                  ;;UPDATE THE POINTER
 4203  017214  023727  016740  016745          CMP     $TKQIN,#$TKQEND         ;;GO OFF THE END?
 4204  017222  001003                          BNE     5$                      ;;BRANCH IF NO
 4205  017224  012737  016744  016740          MOV     #$TKQSRT,$TKQIN         ;;RESET THE POINTER
 4206  017232  000002                  5$:     RTI                             ;;RETURN
 4207
 4208                                          ;;**************************************************************************
 4209                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 4210                                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
 4211                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 4212                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
 4213  017234  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR              ;;IS THE SOFT-SWR SELECTED
 4214  017242  001124                          BNE     15$                     ;;EXIT IF NOT
 4215  017244  105777  161674                  TSTB    @$TKS                   ;;IS A CHAR WAITING?
 4216  017250  100121                          BPL     15$                     ;;IF NOT, EXIT
 4217  017252  117746  161670                  MOVB    @$TKB,-(SP)             ;;YES
 4218  017256  042716  177600                  BIC     #^C177,(SP)             ;;MAKE IT  7-BIT ASCII
 4219  017262  021627  000007                  CMP     (SP),#7                 ;;IS IT A CONTROL-G?
 4220  017266  001300                          BNE     2$                      ;;IF NOT, PUT IT IN THE TTY QUEUE
 4221                                                                          ;;AND EXIT
 4222
 4223                                          ;;**************************************************************************
 4224                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
 4225                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
 4226                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
 4227  017270  123727  001134  000001  6$:     CMPB    $AUTOB,#1               ;;ARE WE RUNNING IN AUTO-MODE?
 4228  017276  001674                          BEQ     2$                      ;;BRANCH IF YES
 4229  017300  005726                          TST     (SP)+                   ;;CLEAR CONTROL-G OFF STACK
 4230  017302  004737  016746                  JSR     PC,$TKINT               ;;FLUSH THE TTY INPUT QUEUE
 4231  017306  005077  161632                  CLR     @$TKS                   ;;DISABLE TTY KEYBOARD INTERRUPTS
 4232  017312  112737  000001  001135          MOVB    #1,$INTAG               ;;SET INTERRUPT MODE INDICATOR
 4233
 4234  017320  104401  020146                  TYPE    ,$CNTLG                 ;;ECHO THE CONTROL-G (^G)
 4235  017324  104401  020153          $GTSWR: TYPE    ,$MSWR                  ;;TYPE CURRENT CONTENTS
 4236  017330  013746  000176                  MOV     SWREG,-(SP)             ;;SAVE SWREG FOR TYPEOUT
 4237  017334  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4238  017336  104401  020164                  TYPE    ,$MNEW                  ;;PROMPT FOR NEW SWR
 4239  017342  005046                  19$:    CLR     -(SP)                   ;;CLEAR COUNTER
 4240  017344  005046                          CLR     -(SP)                   ;;THE NEW SWR
 4241  017346  105777  161572          7$:     TSTB    @$TKS                   ;;CHAR THERE?
 4242  017352  100375                          BPL     7$                      ;;IF NOT TRY AGAIN
 4243
 4244  017354  117746  161566                  MOVB    @$TKB,-(SP)             ;;PICK UP CHAR
 4245  017360  042716  177600                  BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
 4246
 4247  017364  021627  000003                  CMP     (SP),#3                 ;;IS IT A CONTROL-C?
 4248  017370  001015                          BNE     9$                      ;;BRANCH IF NOT
 4249  017372  104401  020134                  TYPE    ,$CNTLC                 ;;YES, ECHO CONTROL-C (^C)
 4250  017376  062706  000006                  ADD     #6,SP                   ;;CLEAN UP STACK
 4251  017402  123727  001135  000001          CMPB    $INTAG,#1               ;;REENABLE TTY KEYBOARD INTERRUPTS?
 4252  017410  001003                          BNE     8$                      ;;BRANCH IF NO
 4253  017412  012777  000100  161524          MOV     #100,@$TKS              ;;ALLOW TTY KEYBOARD INTERRUPTS
 4254  017420  000177  161616          8$:     JMP     @CNTRLC                 ;;CONTROL-C RESTART
```

```
 4255
 4256
 4257  017424  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
 4258  017430  001005                          BNE     10$             ;;BRANCH IF NOT
 4259  017432  104401  020141                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
 4260  017436  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 4261  017442  000737                          BR      19$             ;;LET'S TRY I  AGAIN
 4262
 4263
 4264  017444  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
 4265  017450  001022                          BNE     16$             ;;BRANCH IF NO
 4266  017452  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
 4267  017456  001403                          BEQ     11$             ;;BRANCH IF YES
 4268  017460  016677  000002  161452          MOV     2(SP),@SWR      ;;SAVE NEW SWR
 4269  017466  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
 4270  017472  104401  001213          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
 4271  017476  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
 4272  017504  001003                          BNE     15$             ;;BRANCH IF NOT
 4273  017506  012777  000100  161430          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
 4274  017514  000002          15$:    RTI                             ;;RETURN
 4275  017516  004737  016214          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
 4276  017522  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
 4277  017526  002420                          BLT     18$             ;;BRANCH IF YES
 4278  017530  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 4279  017534  003015                          BGT     18$             ;;BRANCH IF YES
 4280  017536  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 4281  017542  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 4282  017546  001403                          BEQ     17$             ;;BRANCH IF YES
 4283  017550  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 4284  017552  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
 4285  017554  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
 4286  017556  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 4287  017562  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 4288  017566  000667                          BR      7$              ;;GET THE NEXT ONE
 4289  017570  104401  001212          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 4290  017574  000720                          BR      20$             ;;SIMULATE CONTROL-U
 4291                          .DSABL  LSB
 4292
 4293
 4294                          ;;**********************************************************************
 4295                          ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 4296                          ;*CALL:
 4297                          ;*      RDCHR                           ;;GET A CHARACTER FROM THE QUEUE
 4298                          ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
 4299                          ;*                                      ;;WITH PARITY BIT STRIPPED OFF
 4300          :               ;
 4301
 4302  017576  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
 4303  017600  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
 4304  017606  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
 4305  017612  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
 4306  017614  012746  017622                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 4307  017620  000002                          RTI                     ;;POP NEW PC AND PS
 4308  017622                          64$:
 4309  017622  005737  016736          1$:     TST     $TTYCNT         ;;WAIT ON A CHARACTER
 4310  017626  001775                          BEQ     1$
```

```
4311  017630  005337  016736                DEC     $TKCNT              ;;DECREMENT THE COUNTER
4312  017634  117766  177102  000004        MOVB    @$TKQOUT,4(SP)      ;;GET ONE CHARACTER
4313  017642  005237  016742                INC     $TKQOUT             ;;UPDATE THE POINTER
4314  017646  023727  016742  016745        CMP     $TKQOUT,#$TKQEND    ;;DID IT GO OFF OF THE END?
4315  017654  001003                        BNE     2$                  ;;BRANCH IF NO
4316  017656  012737  016744  016742        MOV     #$TKQSRT,$TKQOUT    ;;RESET THE POINTER
4317  017664  000002                2$:     RTI                         ;;RETURN
4318                                        ;;**************************************************************
4319                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4320                                        ;*CALL:
4321                                        ;*      RDLIN                       ;;INPUT A STRING FROM THE TTY
4322                                        ;*      RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4323                                        ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
4324
4325  017666  010346                $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
4326  017670  005046                        CLR     -(SP)               ;;CLEAR THE RUBOUT KEY
4327  017672  012703  020122         $:     MOV     #$TTYIN,R3          ;;GET ADDRESS
4328  017676  022703  020134        2$:     CMP     #$TTYIN+10.,R3      ;;BUFFER FULL?
4329  017702  101456                        BLOS    4$                  ;;BR IF YES
4330  017704  104410                        RDCHP                       ;;GO READ ONE CHARACTER FROM THE TTY
4331  017706  112613                        MOVB    (SP)+,(R3)          ;;GET CHARACTER
4332  017710  122713  000177        10$:    CMPB    #177,(R3)           ;;IS IT A RUBOUT
4333  017714  001022                        BNE     5$                  ;;BR IF NO
4334  017716  005716                        TST     (SP)                ;;IS THIS THE FIRST RUBOUT?
4335  017720  001007                        BNE     6$                  ;;BR IF NO
4336  017722  112737  000134  020120        MOVB    #'\,9$              ;;TYPE A BACK SLASH
4337  017730  104401  020120                TYPE    ,9$
4338  017734  012716  177777                MOV     #-1,(SP)            ;;SET THE RUBOUT KEY
4339  017740  005303                6$:     DEC     R3                  ;;BACKUP BY ONE
4340  017742  020327  020122                CMP     R3,#$TTYIN          ;;STACK EMPTY?
4341  017746  103434                        BLO     4$                  ;;BR IF YES
4342  017750  111337  020120                MOVB    (R3),9$             ;;SETUP TO TYPEOUT THE DELETED CHAR.
4343  017754  104401  020120                TYPE    ,9$                 ;;GO TYPE
4344  017760  000746                        BR      2$                  ;;GO READ ANOTHER CHAR.
4345  017762  005716                5$:     TST     (SP)                ;;RUBOUT KEY SET?
4346  017764  001406                        BEQ     7$                  ;;BR IF NO
4347  017766  112737  000134  020120        MOVB    #'\,9$              ;;TYPE A BACK SLASH
4348  017774  104401  020120                TYPE    ,9$
4349  020000  005016                        CLR     (SP)                ;;CLEAR THE RUBOUT KEY
4350  020002  122713  000025        7$:     CMPB    #25,(R3)            ;;IS CHARACTER A CTRL U?
4351  020006  001003                        BNE     8$                  ;;BR IF NO
4352  020010  104401  020141                TYPE    ,$CNTLU             ;;TYPE A CONTROL 'U''
4353  020014  000726                        BR      1$                  ;;GO START OVER
4354  020016  122713  000022        8$:     CMPB    #22,(R3)            ;;IS CHARACTER A ''^R''?
4355  020022  001011                        BNE     3$                  ;;BRANCH IF NO
4356  020024  105013                        CLRB    (R3)                ;;CLEAR THE CHARACTER
4357  020026  104401  001213                TYPE    ,$CRLF              ;;TYPE A ''CR'' & 'LF''
4358  020032  104401  020122                TYPE    ,$TTYIN             ;;TYPE THE INPUT STRING
4359  020036  000717                        BR      2$                  ;;GO PICKUP ANOTHER CHACTER
4360  020040  104401  001212        4$:     TYPE    ,$QUES              ;;TYPE A '?'
4361  020044  000712                        BR      1$                  ;;CLEAR THE BUFFER AND LOOP
4362  020046  111337  020120        3$:     MOVB    (R3),9$             ;;ECHO THE CHARACTER
4363  020052  104401  020120                TYPE    ,9$
4364  020056  122723  000015                CMPB    #15,(R3)+           ;;CHECK FOR RETURN
4365  020062  001305                        BNE     2$                  ;;LOOP IF NOT RETURN
4366  020064  105063  177777                CLRB    -1(R3)              ;;CLEAR RETURN (THE 15)
```

```
                                        TYPE    ,$LF            ;;TYPE A LINE FEED
                                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
                                        MOV     (SP)+,R3        ;;RESTORE R3
                                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
                                        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
                                        MOV     #$TTYIN,4(SP)
                                        RTI                     ;;RETURN
                                9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
                                        .BYTE   0               ;;TERMINATOR
                                $TTYIN: .BLKB   10.             ;;RESERVE 10. BYTES FOR TTY INPUT
                                $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL 'C'
                                $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
                                $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
                                $MSWR:  .ASCIZ  <15><12>/SWR = /

                                $MNEW:  .ASCIZ  /  NEW = /

                                .EVEN

                                .SBTTL  INTEGER DIVIDE ROUTINE

                                ;;******************************************************************
                                ;*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
                                ;*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
                                ;*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
                                ;*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
                                ;*SAVE SIGN AS THE DIVIDEND.
                                ;*CALL:
                                ;*      MOV     LOW DIVIDEND,-(SP) ;;THE HIGH DIVIDEND MUST BE < 1/2
                                ;*      MOV     HIGH DIVIDEND,-(SP);    AS LARGE AS THE DIVISOR
                                ;*      MOV     DIVISOR,-(SP)
                                ;*      JSR     PC,$DIV
                                ;*      RETURN                  ;;QUOTIENT & REMAINDER ARE ON THE STACK
                                ;*      'V'=0   IMPLIES NO ERROR
                                ;*      'V'=1   IMPLIES ERROR OCCURRED
                                ;*              'C'=0   DIVIDE OVERFLOW OCCURRED
                                ;*              'C'=1   ATTEMPTED TO DIVIDE BY ZERO
                                ;*
                                ;*
                                ;*      STACK   NO ERROR        OVERFLOW        DIVIDE BY ZERO
                                ;*      -----   --------        --------        --------------
                                ;*      TOP     REMAINDER       ALL ZEROS       ALL ONES
                                ;*      +2      QUOTIENT        ALL ZEROS       ALL ONES

                                $DIV:
                                        TRAP                    ;;PUSH OLD PSW AND PC ON STACK
                                        BIC     #17,(SP)        ;;STRIP AWAY CONDITION CODES
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        CLR     -(SP)           ;;SAVE A PLACE FOR SIGNS
                                        MOV     #17.,-(SP)      ;;SETUP THE ITERATION COUNTER
                                        MOV     24(SP),R1       ;;PICKUP THE DIVIDEND
                                        MOV     22(SP),R0
                                        BPL     1$              ;;CHECK THE SIGN
```

```
 4423  020234  105366  000003                DECB    3(SP)           ;;KEEP TRACK OF THE SIGN
 4424  020240  005400                        NEG     R0              ;;AND NEGATE THE ORIGINAL
 4425  020242  005401                        NEG     R1              ;;NUMBER
 4426  020244  005600                        SBC     R0
 4427  020246  016602  000020          1$:   MOV     20(SP),R2       ;;PICKUP THE DIVISOR
 4428  020252  002407                        BLT     2$              ;;CHECK THE SIGN
 4429  020254  003011                        BGT     3$              ;;DIVISOR OF 0 IS A NO-NO
 4430  020256  052766  000003  000014        BIS     #3,14(SP)       ;;SET 'V' & 'C'
 4431  020264  012700  177777                MOV     #-1,R0          ;;SET REMAINDER TO ALL ONES
 4432  020270  000424                        BR      7$              ;;EXIT
 4433  020272  005266  000002          2$:   INC     2(SP)           ;;KEEP TRACK OF DIVISORS SIGN
 4434  020276  000401                        BR      4$
 4435  020300  005402                  3$:   NEG     R2              ;;NEGATE THE ORIGINAL NUMBER
 4436  020302  000241                  4$:   CLC                     ;;CLEAR 'C'
 4437  020304  000405                        BR      6$              ;;START FORMING QUOTIENT
 4438  020306  006100                  5$:   ROL     R0              ;;POSITION MSB'S
 4439  020310  010003                        MOV     R0,R3           ;;COPY
 4440  020312  060203                        ADD     R2,R3           ;;COMPARE DIVIDEND & DIVISOR
 4441  020314  103001                        BCC     6$              ;;BR IF DIVIDEND > DIVISOR
 4442  020316  010300                        MOV     R3,R0           ;;REMAINDER AFTER THIS LOOP
 4443  020320  006101                  6$:   ROL     R1              ;;QUOTIENT BIT ENTERS HERE
 4444  020322  005316                        DEC     (SP)            ;;DONE?
 4445  020324  001370                        BNE     5$              ;;BR IF NO
 4446  020326  005701                        TST     R1              ;;OVERFLOW?
 4447  020330  100005                        BPL     8$              ;;BR IF NO
 4448  020332  052766  000002  000014        BIS     #2,14(SP)       ;;SET 'V' IN RETURN STATUS WORD
 4449  020340  005000                        CLR     R0              ;;SET REMAINDER TO ALL ZEROS
 4450  020342  010001                  7$:   MOV     R0,R1           ;;COPY REMAINDER INTO QUOTIENT
 4451  020344  005726                  8$:   TST     (SP)+           ;;CLEAR COUNTER FROM STACK
 4452  020346  005716                        TST     (SP)            ;;REMAINDER SIGN CORRECTION NEEDED?
 4453  020350  002004                        BGE     9$              ;;BR IF NO
 4454  020352  005400                        NEG     R0              ;;NEGATE REMAINDER
 4455  020354  105066  000001                CLRB    1(SP)           ;;CLEAR SIGN
 4456  020360  005316                        DEC     (SP`            ;;BUT DON'T FORGET QUOTIENT
 4457  020362  005726                  9$:   TST     (SP)+           ;;QUOTIENT SIGN CORRECTION NEEDED?
 4458  020364  001401                        BEQ     10$             ;;BR IF NO
 4459  020366  005401                        NEG     R1              ;;NEGATE QUOTIENT
 4460  020370  010166  000020          10$:  MOV     R1,20(SP)       ;;RETURN QUOTIENT AND
 4461  020374  010066  000016                MOV     R0,16(SP)       ;;REMAINDER TO USER
 4462  020400  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 4463  020402  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4464  020404  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4465  020406  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4466  020410  012666  000002                MOV     (SP)+,2(SP)     ;;SETUP TO RETURN CONDITION CODES
 4467  020414  000002                        RTI                     ;;RETURN
 4468
 4469                                  .SBTTL  SAVF AND RESTORE R0-R5 ROUTINES
 4470
 4471                                  ;;**************************************************************************
 4472                                  ;*SAVE R0-R5
 4473                                  ;*CALL:
 4474                                  ;*      SAVREG
 4475                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
 4476                                  ;*
 4477                                  ;*TOP---(+16)
 4478                                  ;* +2---(+18)
```

```
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0
                                
020416                          $SAVREG:
020416  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020420  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020422  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020424  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020426  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
020430  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020432  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
020436  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
020442  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
020446  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
020452  000002                          RTI
                                
                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
020454                          $RESREG:
020454  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
020460  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
020464  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
020470  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
020474  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
020476  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
020500  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
020502  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
020504  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
020506  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
020510  000002                          RTI
                                
                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
                                
                                ;;**********************************************************************
                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                ;*WITH A RANGE OF 0 TO 2(+33)-1.
                                ;*CALL:
                                ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
                                ;*      RETURN                  ;;RETURN HERE THE RANDOM
                                ;*                              ;;NUMBER WILL BE IN
                                ;*                              ;;$HINUM,$LONUM
                                
020512                          $RAND:
020512  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020514  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020516  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020520  013700  020612                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
020524  013701  020610                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
020530  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
020534  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
020536  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
```

```
 4535  020540  005202                          INC     R2              ;;CHECK FOR DONE
 4536  020542  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
 4537  020544  063700  020612                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
 4538  020550  005501                          ADC     R1              ;;PROPOGATE CARRY
 4539  020552  063701  020610                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
 4540  020556  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
 4541  020562  005501                          ADC     R1              ;;PROPOGATE CARRY
 4542  020564  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
 4543  020570  010037  020612                  MOV     R0,$LONUM       ;;SAVE R0
 4544  020574  010137  020610                  MOV     R1,$HINUM       ;;SAVE R1
 4545  020600  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 4546  020602  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 4547  020604  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 4548  020606  000207                          RTS     PC              ;;RETURN
 4549  020610  176543                $HINUM: .WORD   176543
 4550  020612  123456                $LONUM: .WORD   123456
 4551
 4552                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
 4553
 4554                                  ;;**********************************************************************
 4555                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
 4556                                  ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
 4557                                  ;*POSITIVE.
 4558                                  ;*CALL
 4559                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
 4560                                  ;*      JSR     PC,@#$DB2D
 4561                                  ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
 4562                                                                  ;;IS ON THE STACK
 4563
 4564
 4565  020614  104412                $DB2D:  SAVREG                  ;;SAVE REGISTERS
 4566  020616  016602  000002                MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
 4567  020622  012700  020774                MOV     #$DECVL,R0      ;;GET ADDRESS OF '$DECVL'' STRING
 4568  020626  010066  000002                MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
 4569  020632  012201                        MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
 4570  020634  012202                        MOV     (R2)+,R2
 4571  020636  012737  000012  020712        MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
 4572  020644  012704  020724                MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
 4573  020650  012705  020726                MOV     #$TNPWR+2,R5
 4574  020654  005003                1$:     CLR     R3              ;;CLEAR PARTIAL
 4575  020656  161401                2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
 4576  020660  005602                        SBC     R2
 4577  020662  161502                        SUB     (R5),R2
 4578  020664  002402                        BLT     3$              ;;BR IF TEN POWER TO LARGE
 4579  020666  005203                        INC     R3              ;;ADD 1 TO PARTIAL
 4580  020670  000772                        BR      2$              ;;LOOP
 4581  020672  062401                3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
 4582  020674  005502                        ADC     R2
 4583  020676  062402                        ADD     (R4)+,R2
 4584  020700  022525                        CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
 4585  020702  052703  000060                BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
 4586  020706  110320                        MOVB    R3,(R0)+        ;;SAVE IT
 4587  020710  005327                        DEC     (PC)+           ;;DONE?
 4588  020712  000000                4$:     .WORD   0
 4589  020714  001357                        BNE     1$              ;;BR IF NO
 4590  020716  105020                        CLRB    (R0)+           ;;TERMINATOR
```

DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE SEQ 0085

```
4591  020720  104413                        RESREG                  ;;RESTORE REGISTERS
4592  020722  000207                        RTS     PC              ;;RETURN
4593  020724  145000              $TNPWR:   145000                  ;;1.0E09
4594  020726  035632                        35632
4595  020730  160400                        160400                  ;;1.0E08
4596  020732  002765                        2765
4597  020734  113200                        113200                  ;;1.0E07
4598  020736  000230                        230
4599  020740  041100                        041100                  ;;1.0E06
4600  020742  000017                        17
4601  020744  103240                        103240                  ;;1.0E05
4602  020746  000001                        1
4603  020750  023420                        23420                   ;;1.0E04
4604  020752  000000                        0
4605  020754  001750                        1750                    ;;1.0E03
4606  020756  000000                        0
4607  020760  000144                        144                     ;;1.0E02
4608  020762  000000                        0
4609  020764  000012                        12                      ;;1.0E01
4610  020766  000000                        0
4611  020770  000001                        1                       ;;1.0E00
4612  020772  000000                        0
4613  020774  000014              $DECVL:   .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
4614
4615                                        .SBTTL  TRAP DECODER
4616
4617                                        ;;**********************************************************************
4618                                        ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
4619                                        ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4620                                        ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4621                                        ;*GO TO THAT ROUTINE.
4622
4623  021010  010046              $TRAP:    MOV     R0,-(SP)        ;;SAVE R0
4624  021012  016600  000002               MOV     2(SP),R0        ;;GET TRAP ADDRESS
4625  021016  005740                       TST     -(R0)           ;;BACKUP BY 2
4626  021020  111000                       MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
4627  021022  006300                       ASL     R0              ;;POSITION FOR INDEXING
4628  021024  016000  021044               MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
4629  021030  000200                       RTS     R0              ;;GO TO ROUTINE
4630
4631
4632                                        ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
4633
4634  021032  011646              $TRAP2:   MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
4635  021034  016666  000004  000002       MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
4636  021042  000002                       RTI                     ;;RESTORE THE PSW
4637
4638                                        .SBTTL  TRAP TABLE
4639
4640                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4641                                        ;*BY THE 'TRAP' INSTRUCTION.
4642
4643                                        ;       ROUTINE
4644                                        ;       -------
4645  021044  021032              $TRPAD:   .WORD   $TRAP2
4646  021046  016044                       $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
```

```
4647  021050  016310                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4648  021052  016264                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
4649  021054  016324                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
4650  021056  016512                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
4651
4652  021060  017324                  $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
4653
4654  021062  017234                  $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
4655  021064  017576                  $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
4656  021066  017666                  $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
4657  021070  020416                  $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
4658  021072  020454                  $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
4659
4660                          ;;*****************************************************************
4661
4662                          .SBTTL  RP11 HANDLER
4663
4664                          ;;*****************************************************************
4665
4666                          ;STORAGE FOR RPDS & RPCS ON AN ILLEGAL INTERRUPT ERROR
4667                                  ;RPERRS   - RPDS
4668                                  ;RPERRS+2 = RPCS
4669
4670  021074  000000          RPERRS: .WORD   0
4671  021076  000000                  .WORD   0
4672
4673                          ;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT-1 WORD)
4674                                  ;THIS IS A ONE WORD QUEUE WHICH WILL CONTAIN A
4675                                  ;BIT FOR EACH DRIVE WHICH IS ACTIVE WITH A COMMAND.
4676
4677  021100  000000          DRVACT: .WORD   0
4678
4679                          ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
4680                                  ;DRVSTA=0 DRIVE NONEXISTENT OR OFFLINE
4681                                  ;DRVSTA>0 DRIVE IS ONLINE
4682                                  ;DRVSTA<0 DRIVE IS UNSAFE OR AN RP03
4683
4684  021102      000         DRVSTA: .BYTE   0               ;DRIVE 0
4685  021103      000                 .BYTE   0               ;DRIVE 1
4686  021104      000                 .BYTE   0               ;DRIVE 2
4687  021105      000                 .BYTE   0               ;DRIVE 3
4688  021106      000                 .BYTE   0               ;DRIVE 4
4689  021107      000                 .BYTE   0               ;DRIVE 5
4690  021110      000                 .BYTE   0               ;DRIVE 6
4691  021111      000                 .BYTE   0               ;DRIVE 7
4692
4693                          ;DRIVE TYPE INDICATOR (DRVTYP = 1 WORD)
4694                                  ;THIS IS A ONE WORD INDICATOR.  EACH DRIVE IS ASSIGNED ONE BIT
4695                                  ;STARTING AT BIT00 FOR DRIVE 0.  THE DRIVE'S BIT WILL BE SET IF IT
4696                                  ;IS AN RP03 AND RESET IF IT IS AN RP02.
4697
4698  021112  000000          DRVTYP: .WORD   0
4699
4700                          ;TRANSFER WAIT FLAG (TRNSWT-1 WORD)
4701                                  ;THIS IS A ONE WORD QUEUE.  IT WILL CONTAIN THE ADDRESS OF
4702                                  ;"DPB" OF THE I/O OPERATION.
```

```
4703
4704  021114  000000                  TRNSWT: .WORD   0
4705
4706                                  ;SEEK WAIT KEYS (SEEKWT=1 WORD)
4707                                          ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
4708                                          ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
4709                                          ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
4710                                          ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
4711
4712  021116  000000                  SEEKWT: .WORD   0
4713
4714                                  ;RP11 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
4715                                          ;ACTDRV=0 DRIVER IS INACTIVE
4716                                          ;ACTDRV>0 DRIVER IS ACTIVE
4717
4718  021120      000                 ACTDRV: .BYTE   0
4719
4720                                  ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR-1 BYTE)
4721                                          ;ACTSTR=0 SOFTWARE TIMER ROUTINE IS INACTIVE
4722                                          ;ACTSTR>0 SOFTWARE TIMER ROUTINE IS ACTIVE
4723
4724  021121      000                 ACTSTR: .BYTE   0
4725
4726                                  ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
4727                                          ;SAVEFG<0 SAVE THE RP11E REGISTERS WHEN THE
4728                                          ;OPERATION IS COMPLETED AS PER (DPB+14).
4729                                          ;SAVEFG=0 SAVE THE RP11E REGISTERS, AS PER
4730                                          ;(DPB+14), AFTER AN ERROR.
4731
4732  021122  000000                  SAVEFG: .WORD   0
4733
4734                                  ;ORDER REQUEST QUEUES/LISTS
4735
4736  021124  000000  000000  000000  QREGS:  .WORD   0,0,0,0,0,0,0,0,0  ;STORE REGISTERS ORDER QUEUE
4737  021132  000000  000000  000000
4738  021140  000000  000000  000000
4739  021146  000000  000000  000000  QSEEK:  .WORD   0,0,0,0,0,0,0,0,0  ;SFEK/HOME SEEK ORDER QUEUE
4740  021154  000000  000000  000000
4741  021162  000000  000000  000000
4742  021170  000000  000000  000000  QTRANS: .WORD   0,0,0,0,0,0,0,0,0  ;TRANSFER ORDER QUEUE
4743  021176  000000  000000  000000
4744  021204  000000  000000  000000
4745  021212  000000  000000  000000  QDONE:  .WORD   0,0,0,0,0,0,0,0,0  ;ON CYLINDER TRANSFER QUEUE
4746  021220  000000  000000  000000
4747  021226  000000  000000  000000
4748  021234  000000  000000  000000  QWAIT:  .WORD   0,0,0,0,0,0,0,0    ;TRANSFER SEEK DONE LIST
4749  021242  000000  000000  000000
4750  021250  000000  000000
4751
4752
4753                                  ;TIMEOUT TABLE (TIMER 8 WORDS)
4754                                          ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
4755
4756  021254  177777                  TIMER:  .WORD   -1       ;DRIVE 0
4757  021256  177777                          .WORD   -1       ;DRIVE 1
4758  021260  177777                          .WORD   -1       ;DRIVE 2
```

```
 4759  021262  177777                        .WORD   -1      ;DRIVE 3
 4760  021264  177777                        .WORD   -1      ;DRIVE 4
 4761  021266  177777                        .WORD   -1      ;DRIVE 5
 4762  021270  177777                        .WORD   -1      ;DRIVE 6
 4763  021272  177777                        .WORD   -1      ;DRIVE 7
 4764
 4765                                ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
 4766                                        ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
 4767                                        ;ATTENTION BIT
 4768
 4769  021274      001               ATABIT: .BYTE   1       ;DRIVE 0
 4770  021275      002                       .BYTE   2       ;DRIVE 1
 4771  021276      004                       .BYTE   4       ;DRIVE 2
 4772  021277      010                       .BYTE   10      ;DRIVE 3
 4773  021300      020                       .BYTE   20      ;DRIVE 4
 4774  021301      040                       .BYTE   40      ;DRIVE 5
 4775  021302      100                       .BYTE   100     ;DRIVE 6
 4776  021303      200                       .BYTE   200     ;DRIVE 7
 4777
 4778                                ;RP11E DRIVER INITIALIZATION CODE
 4779                                        ;THIS ROUTINE WILL DETERMINE WHICH RP02 OR RP03 DRIVES ARE
 4780                                        ;AVAILABLE FOR TESTING AND SET THE 'DRVSTA' INDICATOR
 4781                                        ;TO THE PROPER STATE FOR EACH DRIVE.
 4782                                        ;NOTE: THIS ROUTINE CALLS DRVINT
 4783                                ;
 4784                                ;CALL
 4785                                ;
 4786                                ;       JSR     PC,RPINIT
 4787                                ;       RETURN
 4788                                ;
 4789  021304  104412                RPINIT: SAVREG                  ;SAVE R0-R5
 4790  021306  013746  177776                MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
 4791  021312  013737  001222  177776        MOV     RPPRIO,@#PS     ;CHANGE THE PRIORITY
 4792  021320  012701  021074                MOV     #RPERRS,R1      ;FIRST ADDRESS TO BE CLEARED
 4793  021324  012702  021252                MOV     #TIMER-2,R2     ;LAST ADDRESS TO BE CLEARED
 4794  021330  005021                1$:     CLR     (R1)+           ;CLEAR
 4795  021332  020102                        CMP     R1,R2           ;ARE WE DONE?
 4796  021334  101775                        BLOS    1$              ;BRANCH IF NO
 4797  021336  012702  021272                MOV     #ATABIT-2,R2    ;LAST ADDRESS
 4798  021342  012721  177777        2$:     MOV     #-1,(R1)+       ;INITIALIZE
 4799  021346  020102                        CMP     R1,R2           ;DONE?
 4800  021350  101774                        BLOS    2$              ;LOOP IF NO
 4801  021352  013703  001220                MOV     RPVEC,R3        ;SETUP THE RP11 VECTOR
 4802  021356  012723  022472                MOV     #ISR,(R3)+
 4803  021362  013713  001222                MOV     RPPRIO,(R3)
 4804  021366  013704  001216                MOV     RPADR,R4        ;RP11 ADDRESS
 4805  021372  005001                        CLR     R1              ;START WITH DRIVE 0
 4806  021374  004737  021434        3$:     JSR     PC,DRVINT       ;INIT THE DRIVE
 4807  021400  005201                4$:     INC     R1              ;GO TO NEXT DRIVE
 4808  021402  042701  177770                BIC     #^C7,R1         ;MASK OUT UNUSED BITS
 4809  021406  001372                        BNE     3$              ;BR IF MORE DRIVES TO GO
 4810  021410  012701  000007                MOV     #7,R1           ;START WITH DRIVE 7
 4811  021414  005037  177776                CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
 4812  021420  005301                5$:     DEC     R1              ;GO TO THE NEXT DRIVE
 4813  021422  100376                        BPL     5$              ;CHECK NEXT DRIVE
 4814  021424  012637  177776                MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
```

```
 4815  021430  104413                        RESREG                      ;RESTORE R0-R5
 4816  021432  000207                        RTS     PC                  ;BYE-BYE
 4817
 4818                                  ;DRIVE INITIALIZATION ROUTINE
 4819                                  ;THIS ROUTINE DETERMINES OF A DRIVE IS PRESENT AND IF IT IS
 4820                                  ;AN RP02 OR RP03.  'DRVSTA' IS SET TO REFLECT THE DRIVE'S
 4821                                  ;STATUS AND THE DRIVE TYPE INDICATOR BIT IS SET IN 'DRVTYP'.
 4822                                  ;CALL
 4823                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 4824                                  ;       MOV     RPADR,R4        ;UNIBUS ADDRESS OF RP11 (RPDS)
 4825                                  ;       JSR     PC,DRVINT       ;CALLED BY A JSR
 4826                                  ;       RETURN                  ;NORMAL RETURN
 4827                                  ;
 4828  021434  104412                  DRVINT: SAVREG                  ;SAVE R0-R5
 4829  021436  112761  177777  021102          MOVB    #-1,DRVSTA(R1)  ;START DRIVE STATUS AS UNSAFE
 4830  021444  110164  000005                  MOVB    R1,RPCS+1(R4)   ;SELECT A DRIVE
 4831  021450  032764  001000  000000          BIT     #SUFU,RPDS(R4)  ;DRIVE UNSAFE ?
 4832  021456  001022                          BNE     3$              ;BR IF UNSAFE
 4833  021460  032764  040000  000000          BIT     #SUOL,RPDS(R4)  ;DRIVE ONLINE ?
 4834  021466  001410                          BEQ     1$              ;BR IF NOT
 4835  021470  032764  004000  000000          BIT     #SUSI,RPDS(R4)  ;SEEK INCOMPLETE SET ?
 4836  021476  001007                          BNE     2$              ;BR IF SET
 4837  021500  032764  100000  000000          BIT     #SURDY,RPDS(R4)  ;DRIVE READY ?
 4838  021506  001003                          BNE     2$              ;BR IF READY
 4839  021510  105061  021102          1$:     CLRB    DRVSTA(R1)      ;SET DRIVE OFFLINE OR NONEXISTENT
 4840  021514  000416                          BR      5$              ;EXIT
 4841  021516  112761  000001  021102  2$:     MOVB    #1,DRVSTA(R1)   ;SET DRIVE AVAILABLE
 4842  021524  032764  020000  000000  3$:     BIT     #SURP03,RPDS(R4)  ;DRIVE AN RP03 ?
 4843  021532  001004                          BNE     4$              ;BR IF IT IS
 4844  021534  146137  021274  021112          BICB    ATABIT(R1),DRVTYP  ;SET RP02 INDICATOR
 4845  021542  000403                          BR      5$              ;EXIT
 4846  021544  156137  021274  021112  4$:     BISB    ATABIT(R1),DRVTYP  ;SET RP03 INDICATOR
 4847  021552  104413                  5$:     RESREG                  ;RESTORE R0-R5
 4848  021554  000207                          RTS     PC              ;RETURN
 4849
 4850                                  ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
 4851                                  ;
 4852                                  ;CALL
 4853                                  ;
 4854                                  ;       JSR     R0,@#RP11       ;CALL THE RP11E DRIVER
 4855                                  ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVE'S PARAMETER BLOCK
 4856                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
 4857                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
 4858                                  ;                               ;IS AN ERROR CONDITION
 4859
 4860  021556  013746  177776  RP11:   MOV     PSW,-(SP)       ;SAVE THE CALLING STATUS
 4861  021562  013737  001222  177776          MOV     RPPRIO,PSW      ;DON'T ALLOW ANY RP11E INTERRUPTS
 4862  021570  112737  000001  021120          MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
 4863  021576  010246                          MOV     R2,-(SP)        ;SAVE R2
 4864  021600  012002                          MOV     (R0)+,R2        ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
 4865  021602  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
 4866  021606  004037  024050                  JSR     R0,LODQUE       ;PUT THIS REQUEST IN QUEUE
 4867  021612  000403                          BR      1$              ;QUEUE IS FULL
 4868  021614  004737  021636                  JSR     PC,CI           ;INITIATE A COMMAND
 4869  021620  005720                          TST     (R0)+
 4870  021622  012602          1$:     MOV     (SP)+,R2        ;RESTORE R2
```

```
4871  021624  105037  021120                  CLRB    ACTDRV          ;CLEAR ''ACTIVE DRIVER'' FLAG
4872  021630  012637  177776                  MOV     (SP)+,@#PS      ;RETURN 'PS' TO USER LEVEL
4873  021634  000200                          RTS     R0              ;RETURN TO CALLER
4874
4875                                          ;COMMAND INITIATION ROUTINE
4876
4877  021636  104412                  CI:     SAVREG                  ;SAVE R0-R5
4878  021640  013704  001216                  MOV     RPADR,R4        ;RP11 ADDRESS
4879  021644  005737  021114                  TST     TRNSWT          ;TRANSFER IN PROGRESS ?
4880  021650  001035                          BNE     5$              ;BR IF ONE IS
4881  021652  005737  021124                  TST     QREGS           ;PENDING STORE REGISTERS OP ?
4882  021656  001402                          BEQ     1$              ;BR IF NOT
4883  021660  004737  021750                  JSR     PC,CI1          ;STORE THE REGISTERS
4884  021664  005737  021146          1$:     TST     QSEEK           ;PENDING SEEKS OR HOME SEEKS ?
4885  021670  001402                          BEQ     2$              ;BR IF NONE
4886  021672  004737  022014                  JSR     PC,CI2          ;START PENDING SEEKS
4887  021676  005737  021170          2$:     TST     QTRANS          ;PENDING TRANSFER ORDERS ?
4888  021702  001402                          BEQ     3$              ;BR IF NONE
4889  021704  004737  022132                  JSR     PC,CI3          ;START THE TRANSFER ORDERS
4890  021710  005737  021212          3$:     TST     QDONE           ;ON CYLINDER TRANSFER REQUESTS ?
4891  021714  001405                          BEQ     4$              ;BR IF DONE
4892  021716  004737  022254                  JSR     PC,CI4          ;START ORDER AT HEAD OF QUEUE
4893  021722  005737  021114                  TST     TRNSWT          ;TRANSFER ORDER STARTED ?
4894  021726  001006                          BNE     5$              ;BR IF ONE WAS
4895  021730  105737  021100          4$:     TSTB    DRVACT          ;ANY ACTIVE DRIVES ?
4896  021734  001403                          BEQ     5$              ;BR IF NONE
4897  021736  112764  000040  000005          MOVB    #40,RPCS+1(R4)  ;SET ATTENTION INTERRUPT ENABLE
4898  021744  104413                  5$:     RESREG                  ;RESTORE R0-R5
4899  021746  000207                          RTS     PC              ;RETURN
4900
4901                                          ;STORE THE REGISTERS COMMAND
4902
4903  021750  117764  177150  000005  CI1:    MOVB    @QREGS,RPCS+1(R4)  ;SELECT THE DRIVE
4904  021756  013702  021124                  MOV     QREGS,R2        ;DPB ADDRESS
4905  021762  004737  023770                  JSR     PC,SVRP11       ;STORE THE REGISTERS
4906  021766  012762  000200  000016          MOV     #BIT07,16(R2)   ;SET 'DONE'
4907  021774  012702  021124                  MOV     #QREGS,R2       ;QUEUE ADDRESS
4908  022000  004737  024246                  JSR     PC,POPQUE       ;'POP' THE QUEUE
4909  022004  005737  021124                  TST     QREGS           ;ANY MORE REQUESTS IN QUEUE ?
4910  022010  001357                          BNE     CI1             ;BR IF ANY ARE
4911  022012  000207                          RTS     PC              ;RETURN
4912
4913                                          ;START THE SEEK OR HOME SEEK COMMAND
4914
4915  022014  117764  177126  000005  CI2:    MOVB    @QSEEK,RPCS+1(R4)  ;SELECT THE DRIVE
4916  022022  013702  021146                  MOV     QSEEK,R2        ;DPB ADDRESS
4917  022026  005001                          CLR     R1              ;CLEAR R1
4918  022030  117701  177112                  MOVB    @QSEEK,R1       ;DRIVE ADDRESS
4919  022034  004037  022426                  JSR     R0,CIR          ;SEE IF DRIVE IS READY
4920  022040  000424                          BR      1$              ;DRIVE NOT READY OR IS UNSAFE
4921  022042  016264  000010  000012          MOV     10(R2),RPCA(R4)  ;DESIRED CYLINDER ADDRESS
4922  022050  016264  000012  000014          MOV     12(R2),RPDA(R4)  ;SECTOR/TRACK ADDRESS
4923  022056  116264  00C002  000004          MOVB    2(R2),RPCS(R4)  ;START THE COMMAND
4924  022064  156137  021274  021100          BISB    ATABIT(R1),DRVACT  ;SET THE DRIVE'S ACTIVE BIT
4925  022072  006301                          ASL     R1              ;ADDRESS WORDS
4926  022074  013761  001260  021254          MOV     TIMOUT,TIMER(R1)  ;START TIMER FOR DRIVE
```

```
 4927  022102  013761  021146  021234          MOV     QSEEK,QWAIT(R1)  ;PUT DPB ADDRESS IN 'WAIT' QUEUE
 4928  022110  006201                          ASR     R1               ;RESTORE DRIVE ADDPESS
 4929  022112  012701  021146          1$:     MOV     #QSEEK,R1        ;QUEUE ADDRESS
 4930  022116  004737  024246                  JSR     PC,POPQUE        ;'POP' THE SEEK QUEUE
 4931  022122  005737  021146                  TST     QSEEK            ;MORE REQUESTS ?
 4932  022126  001332                          BNE     C12              ;BR IF MORE
 4933  022130  000207                          RTS     PC               ;RETURN
 4934
 4935                                  ;DO AN EXPLICIT SEEK IF DRIVE IS NOT ON CYLINDER
 4936
 4937  022132  117764  177032  000005  C13:     MOVB    @QTRANS,RPCS+1(R4)  ;SELECT THE DRIVE
 4938  022140  005001                          CLR     R1               ;CLEAR R1
 4939  022142  117701  177022                  MOVB    @QTRANS,R1       ;DRIVE ADDRESS
 4940  022146  013702  021170                  MOV     QTRANS,R2        ;DPB ADDRESS
 4941  022152  004037  022426                  JSR     R0,CIR           ;SEE IF DRIVE READY
 4942  022156  000426                          BR      1$               ;DRIVE NOT READY
 4943  022160  016264  000010  000012          MOV     10(R2),RPCA(R4)  ;DESIRED CYLINDER
 4944  022166  016264  000012  000014          MOV     12(R2),RPDA(R4)  ;SECTOR/TRACK ADDRESS
 4945  022174  112764  000011  000004          MOVB    #SEEK,RPCS(R4)   ;SEEK COMMAND
 4946  022202  156137  021274  021100          BISB    ATABIT(R1),DRVACT  ;SET DRIVE ACTIVE
 4947  022210  156137  021274  021116          BISB    ATABIT(R1),SEEKWT  ;SET SEEK WAIT BIT
 4948  022216  006301                          ASL     R1               ;SETUP TO ADDRESS WORDS
 4949  022220  013761  001260  021254          MOV     TIMOUT,TIMER(R1)  ;START TIMER FOR DRIVE
 4950  022226  013761  021170  021234          MOV     QTRANS,QWAIT(R1)  ;'WAIT' TABLE ENTRY
 4951  022234  012701  021170          1$:     MOV     #QTRANS,R1       ;TRANSFER QUEUE ADDRESS
 4952  022240  004737  024246                  JSR     PC,POPQUE        ;'POP' THE TRANSFER QUEUE
 4953  022244  005737  021170                  TST     QTRANS           ;MORE DRIVES IN QUEUE ?
 4954  022250  001330                          BNE     C13              ;BR IF MORE DRIVES
 4955  022252  000207                          RTS     PC               ;RETURN
 4956
 4957                                  ;START THE PENDING ON CYLINDER TRANSFER REQUEST
 4958
 4959  022254  117764  176732  000005  C14:     MOVB    @QDONE,RPCS+1(R4)  ;SELECT THE DRIVE
 4960  022262  117701  176724                  MOVB    @QDONE,R1        ;DRIVE ADDRESS
 4961  022266  013702  021212                  MOV     QDONE,R2         ;DPB ADDRESS
 4962  022272  004037  022426                  JSR     R0,CIR           ;SEE IF DRIVE IS READY
 4963  022276  000440                          BR      1$               ;DRIVF NOT READY
 4964  022300  016264  000004  000006          MOV     4(R2),RPWC(R4)   ;WORD COUNT
 4965  022306  016264  000006  000010          MOV     6(R2),RPBA(R4)   ;BUFFER ADDRESS
 4966  022314  016264  000010  000012          MOV     10(R2),RPCA(R4)  ;CYLINDER ADDRESS
 4967  022322  016264  000012  000014          MOV     12(R2),RPDA(R4)  ;SECTOR/TRACK ADDRESS
 4968  022330  016246  000002                  MOV     2(R2),-(SP)      ;COMMAND CODE, MODE & HDR BITS
 4969  022334  151266  000001                  BISB    (R2),1(SP)       ;DRIVE ADDRESS
 4970  022340  156216  000001                  BISB    1(R2),(SP)       ;'MEX' BITS
 4971  022344  052716  000100                  BIS     #IDE,(SP)        ;INTERRUPT ON DONE BIT
 4972  022350  012664  000004                  MOV     (SP)+,RPCS(R4)   ;START THE COMMAND
 4973  022354  156137  021274  021100          BISB    ATABIT(R1),DRVACT  ;DRIVE ACTIVE BIT
 4974  022362  006301                          ASL     R1               ;ADDRESS WORDS
 4975  022364  013761  001260  021254          MOV     TIMOUT,TIMER(R1)  ;START TIMER FOR DRIVE
 4976  022372  006201                          ASR     R1               ;RESTORE R1
 4977  022374  010237  021114                  MOV     R2,TRNSWT        ;DPB ADDRESS
 4978  022400  012701  021212          1$:     MOV     #QDONE,R1        ;QUEUE ADDRESS
 4979  022404  004737  024246                  JSR     PC,POPQUE        ;'POP' THE QUEUE
 4980  022410  005737  021114                  TST     TRNSWT           ;TRANSFER ORDER RUNNING ?
 4981  022414  001003                          BNE     2$               ;BR IF ONE IS
 4982  022416  005737  021212                  TST     QDONE            ;MORE ORDERS WAITING ?
```

```
 4983  022422  001314                          BNE     C14             ;BR IF MORE
 4984  022424  000207                  2$:     RTS     PC              ;RETURN
 4985
 4986                                  ;SEE IF DRIVE IS ONLINE
 4987
 4988  022426  004737  021434          CIR:    JSR     PC,DRVINT       ;CHECK DRIVE'S STATUS
 4989  022432  105761  021102                  TSTB    DRVSTA(R1)      ;SEE WHAT DRIVE'S STATUS IS
 4990  022436  003013                          BGT     3$              ;BR IF DRIVE ONLINE
 4991  022440  002404                          BLT     1$              ;BR IF DRIVE IS UNSAFE
 4992  022442  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE INDICATOR
 4993  022450  000403                          BR      2$              ;GO STORE REGISTERS
 4994  022452  012762  110000  000016  1$:     MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE INDICATOR
 4995  022460  004737  023770          2$:     JSR     PC,SVRP11       ;STORE THE REGISTERS
 4996  022464  000401                          BR      4$              ;EXIT
 4997  022466  005720                  3$:     TST     (R0)+           ;STEP AROUND ERROR RETURN
 4998  022470  000200                  4$:     RTS     R0              ;RETURN
 4999
 5000                                  ;INTERRUPT SERVICE ROUTINE
 5001
 5002  022472  112737  000001  021120  ISR:    MOVB    #1,ACTDRV       ;SET ''ACTIVE DRIVER'' FLAG
 5003  022500  104412                          SAVREG                  ;SAVE R0-R5
 5004  022502  013704  001216                  MOV     RPADR,R4        ;ADDRESS OF RP11
 5005  022506  005737  021114                  TST     TRNSWT          ;TRANSFER OPERATION IN PROGRESS ?
 5006  022512  001403                          BEQ     1$              ;BR IF NOT
 5007  022514  004737  022542                  JSR     PC,TD           ;CALL TRANSFER DONE
 5008  022520  000402                          BR      2$              ;EXIT
 5009  022522  004737  023024          1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
 5010  022526  004737  021636          2$:     JSR     PC,CI           ;GO TO THE COMMAND INITIATION ROUTINE
 5011  022532  104413                          RESREG                  ;RESTORE R0-R5
 5012  022534  105037  021120                  CLRB    ACTDRV          ;CLEAR ''ACTIVE DRIVER'' FLAG
 5013  022540  000002                          RTI                     ;RETURN
 5014
 5015                                  ;TRANSFER DONE ROUTINE
 5016
 5017  022542  005001                  TD:     CLR     R1              ;CLEAR DRIVE NUMBER STORAGE
 5018  022544  117701  176344                  MOVB    @TRNSWT,R1      ;GET DRIVE NUMBER
 5019  022550  146137  021274  021100          BICB    ATABIT(R1),DRVACT  ;CLEAR DRIVE ACTIVE BIT
 5020  022556  006301                          ASL     R1
 5021  022560  012761  177777  021254          MOV     #-1,TIMER(R1)   ;STOP TIMER
 5022  022566  006201                          ASR     R1
 5023  022570  013702  021114                  MOV     TRNSWT,R2       ;GET ''DPB'' ADDRESS FROM THE
 5024  022574  005037  021114                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
 5025  022600  012762  000200  000016          MOV     #BIT07,16(R2)   ;SET DONE
 5026  022606  110164  000005                  MOVB    R1,RPCS+1(R4)   ;SELECT THE DRIVE
 5027  022612  116164  021274  000000          MOVB    ATABIT(R1),RPDS(R4)  ;CLEAR DRIVE'S ATTENTION BIT
 5028  022620  005764  000004                  TST     RPCS(R4)        ;ERROR SET ?
 5029  022624  100413                          BMI     2$              ;BR IF YES
 5030  022626  005737  021122                  TST     SAVEFG          ;SAVE THE RP11E REGISTERS?
 5031  022632  100002                          BPL     1$              ;BRANCH IF NO
 5032  022634  004737  023770                  JSR     PC,SVRP11       ;YES--SAVE THE REGISTERS
 5033  022640  042764  000100  000004  1$:     BIC     #IDE,RPCS(R4)   ;CLEAR 'INTERRUPT ON DONE' BIT
 5034  022646  004737  022732                  JSR     PC,WC           ;SEE IF WRITE CHECK TO BE PUT IN QUEUE
 5035  022652  000505                          BR      SC0             ;SPECIAL CONDITIONS ENTRY
 5036  022654  004737  023770          2$:     JSR     PC,SVRP11       ;SAVE THE REGISTERS
 5037  022660  012764  000001  000004          MOV     #CLEAR,RPCS(R4)  ;CLEAR THE CONTROLLER
 5038  022666  004737  021434                  JSR     PC,DRVINT       ;CHECK THE DRIVE'S STATUS
```

```
5039  022672  105761  021102                    TSTB    DRVSTA(R1)          ;DRIVE STILL ONLINE ?
5040  022676  003011                            BGT     4$                  ;BR IF ONLINE
5041  022700  100404                            BMI     3$                  ;BR IF UNSAFE
5042  022702  012762  140000  000016            MOV     #BIT15!BIT14,16(R2)  ;SET DRIVE OFFLINE INDICATOR
5043  022710  000466                            BR      SC0                 ;CHECK FOR OTHER DRIVES
5044  022712  012762  110000  000016  3$:       MOV     #BIT15!BIT12,16(R2)  ;SET DRIVE UNSAFE INDICATOR
5045  022720  000462                            BR      SC0                 ;CHECK FOR OTHER DRIVES
5046  022722  052762  100100  000016  4$:       BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
```

```
5047  022730  000456                          BR      SC0             ;SPECIAL CONDITIONS ENTRY
5048
5049                                  ;FORCED WRITE CHECK ROUTINE
5050
5051  022732  005737  001322          WC:     TST     AUTOCK          ;AUTOMATIC WRITE CHECKS ?
5052  022736  001430                          BEQ     1$              ;BR IF NOT
5053  022740  132762  000004  000002          BITB    #BIT02,$COMND(R2)  ;WRITE COMMAND ?
5054  022746  001024                          BNE     1$              ;BR IF NOT
5055  022750  010146                          MOV     R1,-(SP)        ;SAVE R1
5056  022752  012701  021170                  MOV     #QTRANS,R1      ;TRANSFER QUEUE ADDRESS
5057  022756  004037  024216                  JSR     R0,DRVQUE       ;PUT THE OPERATION IN THE QUEUE
5058  022762  000416                          BR      1$              ;QUEUE IS FULL
5059  022764  005062  000016                  CLR     16(R2)          ;CLEAR 'DONE' BIT IN DPB
5060  022770  116262  000002  000024          MOVB    $COMND(R2),$PREVO(R2)  ;SAVE WRITE OPERATION CODE
5061  022776  016262  000010  000032          MOV     $CYL(R2),$PREVA+2(R2)  ;SAVE CYLINDER
5062  023004  016262  000012  000030          MOV     $SEC(R2),$PREVA(R2)  ;SAVE SECTOR AND TRACK ADDRESSES
5063  023012  112762  000007  000002          MOVB    #WRTCHK,$COMND(R2)  ;CHANGE DRIVER CODE TO WRITE CHECK
5064  023020  012601                  1$:     MOV     (SP)+,R1        ;RESTORE R1
5065  023022  000207                          RTS     PC              ;EXIT
5066
5067                                  ;SPECIAL CONDITION ROUTINE
5068
5069  023024  105764  000000          SC:     TSTB    RPDS(R4)        ;ATTENTION BITS SET ?
5070  023030  001016                          BNE     SC0             ;BR IF SOME SET
5071  023032  016437  000000  021074          MOV     RPDS(R4),RPERRS ;STORE DRIVE STATUS REGITER
5072  023040  016437  000004  021076          MOV     RPCS(R4),RPERRS+2  ;STORE THE CONTROL REGISTER
5073  023046  104003                          ERROR   3               ;ILLEGAL INTERRUPT
5074  023050  005737  021100                  TST     DRVACT          ;ANY DRIVES ACTIVE ?
5075  023054  001403                          BEQ     1$              ;BR IF NOT
5076  023056  112764  000040  000005          MOVB    #40,RPCS+1(R4)  ;SET 'AIE'
5077  023064  000207                  1$:     RTS     PC              ;RETURN
5078
5079
5080  023066  005001                  SC0:    CLR     R1              ;CLEAR DRIVE NUMBER STORAGE
5081  023070  012703  000001                  MOV     #1,R3           ;START WITH DRIVE 0'S BIT
5082  023074  030337  021100          SC1:    BIT     R3,DRVACT       ;IS THIS DRIVE ACTIVE ?
5083  023100  001004                          BNE     SC3             ;BR IF IT IS
5084  023102  005201                  SC2:    INC     R1              ;GO TO NEXT DRIVE
5085  023104  106303                          ASLB    R3              ;SHIFT DRIVE'S BIT
5086  023106  001372                          BNE     SC1             ;BR IF MORE DRIVES TO CHECK
5087  023110  000207                          RTS     PC              ;RETURN
5088  023112  006301                  SC3:    ASL     R1              ;SETUP TO ADDRESS WORDS
5089  023114  016102  021234                  MOV     QWAIT(R1),R2    ;GET DRIVE'S DPB ADDRESS
5090  023120  006201                          ASR     R1              ;RESTORE ADDRESS
5091  023122  110164  000005                  MOVB    R1,RPCS+1(R4)   ;SELECT DRIVE
5092  023126  032764  002000  000000          BIT     #SUSU,RPDS(R4)  ;DRIVE STILL SEEKING ?
5093  023134  001362                          BNE     SC2             ;BR IF IT IS
5094  023136  006301                          ASL     R1              ;ADDRESS WORDS
5095  023140  005061  021234                  CLR     QWAIT(R1)       ;CLEAR THE WAIT QUEUE ENTRY
5096  023144  012761  177777  021254          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
5097  023152  006201                          ASR     R1              ;RESTORE DRIVE ADDRESS
5098  023154  146137  021274  021100          BICB    ATABIT(R1),DRVACT  ;CLEAR THE DRIVE'S ACTIVE BIT
5099  023162  116164  021274  000000          MOVB    ATABIT(R1),RPDS(R4)  ;CLEAR DRIVE'S ATTENTION BIT
5100  023170  032764  001000  000000          BIT     #SUFU,RPDS(R4)  ;DRIVE UNSAFE ?
5101  023176  001027                          BNE     1$              ;BR IF IT IS
5102  023200  032764  040000  000000          BIT     #SUOL,RPDS(R4)  ;IS DRIVE STILL ONLINE ?
```

```
5103  023206  001432                          BEQ     2$                  ;BR IF NOT
5104  023210  032764  140000  000004          BIT     #ERR.HE,RPCS(R4)    ;ANY ERRORS SET ?
5105  023216  001034                          BNE     3$                  ;BR IF SET
5106  023220  136137  021274  021116          BITB    ATABIT(R1),SEEKWT   ;TRANSFER SEEK ?
5107  023226  001434                          BEQ     4$                  ;BR IF NOT
5108  023230  010146                          MOV     R1,-(SP)            ;SAVE R1
5109  023232  012701  021212                  MOV     #QDONE,R1           ;DONE QUEUE ADDRESS
5110  023236  004037  024216                  JSR     R0,DRVQUE           ;PUT DRIVE'S DPB ADDRESS IN DONE QUEUE
5111  023242  000240                          NOP                         ;RETURN IF QUEUE FULL (SHOULDN'T HAPPEN)
5112  023244  012601                          MOV     (SP)+,R1            ;RESTORE R1
5113  023246  146137  021274  021116          BICB    ATABIT(R1),SEEKWT   ;CLEAR WAIT BIT
5114  023254  000712                          BR      SC2                 ;GET ANOTHER DRIVE
5115  023256  052762  110000  000016  1$:     BIS     #BIT15!BIT12,16(R2) ;SET DRIVE UNSAFE STATUS
5116  023264  112761  177777  021102          MOVB    #-1,DRVSTA(R1)      ;SET DRIVE UNSAFE
5117  023272  000420                          BR      5$                  ;STORE REGISTERS
5118  023274  052762  140000  000016  2$:     BIS     #BIT15!BIT14,16(R2) ;SET DRIVE OFFLINE STATUS
5119  023302  105061  021102                  CLRB    DRVSTA(R1)          ;SET DRIVE OFFLINE
5120  023306  000412                          BR      5$                  ;STORE REGISTERS
5121  023310  052762  100240  000016  3$:     BIS     #BIT15.BIT7!BIT5,16(R2) ;SET ERROR DURING SEEK STATUS
5122  023316  000406                          BR      5$                  ;STORE REGISTERS
5123  023320  052762  000200  000016  4$:     BIS     #BIT07,16(R2)       ;NORMAL END OF ORDER STATUS
5124  023326  005737  021122                  TST     SAVEFG              ;SAVE REGISTERS ?
5125  023332  001663                          BEQ     SC2                 ;BR IF NOT
5126  023334  146137  021274  021116  5$:     BICB    ATABIT(R1),SEEKWT   ;CLEAR DRIVE'S WAIT BIT
5127  023342  004737  023770                  JSR     PC,SVRP11           ;STORE THE RP11E REGISTERS
5128  023346  000655                          BR      SC2                 ;GET ANOTHER DRIVE
5129
5130                                  ;TIMER ROUTINE
5131                                  ;CALL
5132                                  ;       MOV     #TIME,-(SP)         ;ELASPED TIME IN MILLISECONDS ON THE STACK
5133                                  ;       JSR     PC,RPTMR            ;CALL RP11E TIMER ROUTINE
5134
5135  023350  005737  021120          RPTMR:  TST     ACTDRV              ;CHECK ''ACTDRV & ACTSTR''
5136  023354  001040                          BNE     5$                  ;IF NON-ZERO, EXIT
5137  023356  112737  000001  021121          MOVB    #1,ACTSTR           ;SET ''ACTSTR''
5138  023364  104412                          SAVREG                      ;SAVE R0-R5
5139  023366  005001                          CLR     R1                  ;START WITH DRIVE 0
5140  023370  005003                          CLR     R3
5141  023372  005763  021254          1$:     TST     TIMER(R3)           ;IS THE TIMER RUNNING?
5142  023376  002407                          BLT     2$                  ;BRANCH IF NO
5143  023400  166663  000002  021254          SUB     2(SP),TIMER(R3)     ;COUNT THE INTERVAL
5144  023406  003003                          BGT     2$                  ;BR IF NO SOFTWARE TIMEOUT
5145  023410  004737  023462                  JSR     PC,STO              ;CALL SOFTWARE TIMEOUT ROUTINE
5146  023414  000406                          BR      3$                  ;GO TO THE EXIT
5147  023416  005201                  2$:     INC     R1                  ;MOVE TO NEXT DRIVE
5148  023420  005723                          TST     (R3)+
5149  023422  022701  000010                  CMP     #8.,R1              ;OUT OF DRIVES?
5150  023426  003361                          BGT     1$                  ;BRANCH IF NO
5151  023430  000407                          BR      4$                  ;BYPASS 'SC' AND 'CI' CHECKS
5152  023432  005737  021114          3$:     TST     TRNSWT              ;ACTIVE TRANSFER COMMAND ?
5153  023436  001004                          BNE     4$                  ;BR IF YES
5154  023440  004737  023066                  JSR     PC,SCO              ;LOOK FOR DRIVES FINISHED SEEKING
5155  023444  004737  021636                  JSR     PC,CI               ;START ANY OUTSTANDING COMMANDS
5156  023450  104413                  4$:     RESREG                      ;RESTORE R0-R5
5157  023452  105037  021121                  CLRB    ACTSTR              ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
5158  023456  012616                  5$:     MOV     (SP)+,(SP)          ;ADJUST THE STACK
```

```
5159  023460  000207                          RTS     PC              ;RETURN
5160
5161                                  ;SOFTWARE TIMEOUT ROUTINE
5162                                  ;
5163                                  ;CALL:  STO
5164                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
5165                                  ;       JSR     PC,STO          ;CALL--DRVACT MUST BE NONZERO
5166                                  ;       RETURN
5167
5168  023462  013704  001216          STO:    MOV     RPADR,R4        ;GET ADDRESS OF RP11
5169  023466  110164  000005                  MOVB    R1,RPCS+1(R4)   ;SELECT THE DRIVE
5170  023472  006301                          ASL     R1              ;WORD INDEX
5171  023474  012761  177777  021254          MOV     #-1,TIMER(R1)   ;STOP TIMER
5172  023502  006201                          ASR     R1              ;RESTORE R1
5173  023504  005737  021114                  TST     TRNSWT          ;DATA TRANSFER IN PROGRESS ?
5174  023510  001446                          BEQ     STO1            ;BR IF NOT
5175  023512  120177  175376                  CMPB    R1,@TRNSWT      ;IN PROGRESS FOR THIS DRIVE ?
5176  023516  001123                          BNE     STO5            ;BR IF NOT
5177  023520  105764  000004                  TSTB    RPCS(R4)        ;CONTROLLER READY ?
5178  023524  100520                          BMI     STO5            ;BR IF READY
5179  023526  146137  021274  021100          BICB    ATABIT(R1),DRVACT  ;CLEAR DRIVE ACTIVE BIT
5180  023534  013702  021114                  MOV     TRNSWT,R2       ;PUT DPB ADDRESS IN R2
5181  023540  004737  023770                  JSR     PC,SVRP11       ;STORE THE REGISTERS
5182  023544  004737  021434                  JSR     PC,DRVINT       ;CHECK ON THE DRIVE'S STATUS
5183  023550  012762  100400  000016          MOV     #BIT15!BIT08,16(R2)  ;SET TIMEOUT & 'RESET' INDICATOR
5184  023556  105761  021102                  TSTB    DRVSTA(R1)      ;SEE IF DRIVE STILL ONLINE
5185  023562  003010                          BGT     2$              ;BR IF DRIVE STILL ONLINE
5186  023564  002404                          BLT     1$              ;BR IF DRIVE UNSAFE
5187  023566  052762  040000  000016          BIS     #BIT14,16(R2)   ;SET OFFLINE INDICATOR
5188  023574  000403                          BR      2$              ;GO INIT THE DRIVE
5189  023576  052762  010000  000016  1$:     BIS     #BIT12,16(R2)   ;SET UNSAFE INDICATOR
5190  023604  112764  000001  000004  2$:     MOVB    #CLEAR,RPCS(R4) ;CLEAR THE CONTROLLER
5191  023612  112764  000001  000004          MOVB    #CLEAR,RPCS(R4) ;CLEAR THE CONTROLLER AGAIN
5192  023620  005037  021114                  CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
5193  023624  000460                          BR      STO5            ;EXIT
5194  023626  006301                  STO1:   ASL     R1              ;ADDRESS WORDS
5195  023630  016102  021234                  MOV     QWAIT(R1),R2    ;GET DPB ADDRESS
5196  023634  006201                          ASR     R1              ;RESTORE R1
5197  023636  032764  001000  000000          BIT     #SUFU,RPDS(R4)  ;DRIVE UNSAFE ?
5198  023644  001020                          BNE     STO2            ;BR IF UNSAFE
5199  023646  032764  004000  000000          BIT     #SUSI,RPDS(R4)  ;SEEK INCOMPLETE ?
5200  023654  001044                          BNE     STO5            ;BR IF SET
5201  023656  032764  040000  000000          BIT     #SUOL,RPDS(R4)  ;DRIVE ONLINE ?
5202  023664  001417                          BEQ     STO3            ;BR IF NOT
5203  023666  032764  100000  000000          BIT     #SURDY,RPDS(R4) ;DRIVE READY ?
5204  023674  001034                          BNE     STO5            ;BR IF READY
5205  023676  012762  101000  000016          MOV     #BIT15!BIT09,16(R2)  ;SET INDICATOR FOR TIMEOUT DURING SEEK
5206  023704  000414                          BR      STO4            ;CONTINUE WITH PROCESSING
5207  023706  012762  110000  000016  STO2:   MOV     #BIT15!BIT12,16(R2)  ;DRIVE UNSAFE INDICATOR
5208  023714  112761  177777  021102          MOVB    #-1,DRVSTA(R1)  ;SET DRIVE UNSAFE
5209  023722  000405                          BR      STO4            ;CONTINUE
5210  023724  012762  140000  000016  STO3:   MOV     #BIT15.BIT14,16(R2)  ;DRIVE OFFLINE INDICATOR
5211  023732  105061  021102                  CLRB    DRVSTA(R1)      ;SET DRIVE OFFLINE
5212  023736  006301                  STO4:   ASL     R1              ;ADDRESS WORDS
5213  023740  005061  021234                  CLR     QWAIT(R1)       ;CLEAR THE ENTRY
5214  023744  006201                          ASR     R1              ;RESTORE R1
```

```
                                        BICB    ATABIT(R1),DRVACT  ;CLEAR DRIVE'S ACTIVE BIT
                                        BICB    ATABIT(R1),SEEKWT  ;CLEAR SEEK WAIT BIT
                                        JSR     PC,SVRP11          ;SAVE THE REGISTERS
                                STO5:   RTS     PC                 ;RETURN


                                ;ROUTINE TO SAVE THE RP11E REGISTERS AS PER DPB+14
                                ;
                                ;CALL
                                ;       MOV     #DPBNUM,R2         ;DPB POINTER TO R2
                                ;       JSR     PC,SVRP11          ;SAVE THE DRIVES REG'S

023770  104412                  SVRP11: SAVREG                     ;SAVE R0-R5
023772  013704  001216                  MOV     RPADR,R4           ;RP11 ADDRESS
023776  111264  000005                  MOVB    (R2),RPCS+1(R4)    ;SELECT DRIVE
024002  016202  000014                  MOV     14(R2),R2          ;GET THE ERROR TABLE POINTER
024006  001416                          BEQ     4$                 ;EXIT IF 0
024010  005003                          CLR     R3                 ;COUNTER & POINTER
024012  012705  000020                  MOV     #RPM2,R5           ;PROBLEM REGISTER
024016  020305                  1$:     CMP     R3,R5              ;REACHED RPM2 ?
024020  001002                          BNE     2$                 ;BR IF NO
024022  005723                          TST     (R3)+              ;INCREMENT THE INDEX
024024  005723                          TST     (R3)+              ;INCREMENT THE INDEX
024026  010446                  2$:     MOV     R4,-(SP)           ;FORM RP11 ADDRESS THAT IS
024030  060316                          ADD     R3,(SP)            ;TO BE READ
024032  013622                          MOV     @(SP)+,(R2)+       ;AND READ IT
024034  005723                  3$:     TST     (R3)+              ;MOVE TO NEXT REG INDEX
024036  020327  000026                  CMP     R3,#SILO           ;DONE?
024042  003765                          BLE     1$                 ;BRANCH IF NO
024044  104413                  4$:     RESREG                     ;RESTORE R0-R5
024046  000207                          RTS     PC                 ;RETURN TO USER


                                ;ROUTINE TO PUT THE REQUEST IN THE PROPER QUEUE
                                ;
                                ;CALL
                                ;       MOV     #DPBNUM,R2         ;ADDRESS OF PARAMETER BLOCK
                                ;       JSR     R0,LODQUE          ;PUT THE REQUEST IN THE PROPER QUEUE
                                ;       RETURN1                    ;RETURN HERE IF QUEUE IF FULL
                                ;       RETURN2                    ;RETURN HERE IF REQUEST IS IN QUEUE

024050  010146                  LODQUE: MOV     R1,-(SP)           ;SAVE R1
024052  032762  000001  000002          BIT     #BIT0,2(R2)        ;IS THIS A GET REGISTER REQUEST ?
024060  001003                          BNE     1$                 ;BR IF NOT
024062  012701  021124                  MOV     #QREGS,R1          ;LOAD QUEUE ADDRESS
024066  000445                          BR      7$                 ;GO ENTER REQUEST
024070  122762  000011  000002  1$:     CMPB    #SEEK,2(R2)        ;SEEK REQUEST ?
024076  001404                          BEQ     2$                 ;BR IF IT IS
024100  122762  000015  000002          CMPB    #HOMSEK,2(R2)      ;HOME SEEK REQUEST ?
024106  001003                          BNE     3$                 ;BR IF NOT
024110  012701  021146          2$:     MOV     #QSEEK,R1          ;QUEUE ADDRESS
024114  000432                          BR      7$                 ;ENTER REQUEST
024116  122762  000005  000002  3$:     CMPB    #RDSEK,2(R2)       ;READ W/IMPLIED SEEK ?
024124  001424                          BEQ     5$                 ;BR IF IT IS
024126  122762  000003  000002          CMPB    #WRTSEK,2(R2)      ;WRITE W/IMPLIED SEEK ?
024134  001420                          BEQ     5$                 ;BR IF IT IS
024136  122762  000007  000002          CMPB    #WRTCHK,2(R2)      ;WRITE CHECK ?
```

```
5271  024144  001414                        BEQ     5$              ;BR IF WRITE CHECK
5272  024146  122762  000017  000002        CMPB    #READ,2(R2)     ;READ W/O IMPLIED SEEK ?
5273  024154  001410                        BEQ     5$              ;BR IF IT IS
5274  024156  122762  000013  000002        CMPB    #WRITE,2(R2)    ;WRITE W/O IMPLIED SEEK ?
5275  024164  001404                        BEQ     5$              ;BR IF IT IS
5276  024166  012762  120000  000016        MOV     #BIT15!BIT13,16(R2)  ;SET INVALID OPCODE INDICATOR
5277  024174  000405                        BR      8$              ;EXIT WITHOUT MAKING QUEUE ENTRY
5278  024176  012701  021170          5$:   MOV     #QTRANS,R1      ;QUEUE ADDRESS
5279  024202  004037  024216          7$:   JSR     R0,DRVQUE       ;PUT THE REQUEST IN THE QUEUE
5280  024206  000401                        BR      9$              ;QUEUE WAS FULL
5281  024210  005720                  8$:   TST     (R0)+           ;INCREMENT RETURN
5282  024212  012601                  9$:   MOV     (SP)+,R1        ;RESTORE R1
5283  024214  000200                        RTS     R0              ;RETURN
5284
5285                                  ;ROUTINE TO PUT A REQUEST IN QUEUE
5286                                  ;
5287                                  ;CALL
5288                                  ;       MOV     #QNAME,R1       ;QUEUE ADDRESS
5289                                  ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
5290                                  ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
5291                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
5292                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
5293
5294  024216  012746  000010          DRVQUE: MOV     #8.,-(SP)       ;QUEUE SIZE COUNTER
5295  024222  005721                  1$:     TST     (R1)+           ;SEE IF AT END OF ENTRIES
5296  024224  001403                          BEQ     2$              ;BR IF YES
5297  024226  005316                          DEC     (SP)            ;DECREMENT THE COUNTER
5298  024230  003374                          BGT     1$              ;BR IF NOT AT END
5299  024232  000403                          BR      3$              ;AT END
5300  024234  010261  177776          2$:     MOV     R2,-2(R1)       ;PUT REQUEST IN THE QUEUE
5301  024240  005720                          TST     (R0)+           ;INCREMENT RETURN
5302  024242  005726                  3$:     TST     (SP)+           ;CORRECT THE STACK POINTER
5303  024244  000200                          RTS     R0              ;RETURN
5304
5305                                  ;ROUTINE TO 'POP' THE REQUEST FROM QUEUE
5306                                  ;
5307                                  ;CALL
5308                                  ;       MOV     #QNAME,R1       ;QUEUE ADDRESS TO R1
5309                                  ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
5310
5311  024246  016111  000002          POPQUE: MOV     2(R1),(R1)      ;SHIFT ENTRY DOWN
5312  024252  001403                          BEQ     1$              ;BR IF AT END
5313  024254  062701  000002                  ADD     #2,R1           ;INCREMENT THE POINTER
5314  024260  000772                          BR      POPQUE          ;CONTINUE
5315  024262  000207                  1$:     RTS     PC              ;RETURN
5316
5317                                  ;;*****************************************************************
5318
5319                                  .SBTTL  DATA, CONTROL, & STATUS BLOCKS
5320
5321                                  ;;*****************************************************************
5322
5323                                  ;BLOCK LOCATION EQUATE STATEMENTS
5324
5325          000001                  $MEX    =       1               ;EXTENDED MEMORY BITS
5326          000002                  $COMND  =       $MEX+1          ;OPERATION CODE
```

```
 5327          000003                  $MODE   =       $COMND+1        ;MODE & HDR BITS
 5328          000004                  $WRDM   =       $MODE+1         ;WORD COUNT (2'S COMP)
 5329          000006                  $BUF    =       $WRDM+2         ;BUFFER ADDR
 5330          000010                  $CYL    =       $BUF+2          ;CYLINDER ADDRESS
 5331          000012                  $SEC    =       $CYL+2          ;SECTOR ADDRESS
 5332          000013                  $TRK    =       $SEC+1          ;TRACK ADDRESS OF LAST REG ADDR
 5333          000014                  $REG    =       $TRK+1          ;REGISTER STORAGE ADDRESS
 5334          000016                  $TATUS  =       $REG+2          ;STATUS WORD (SET BY DRIVER)
 5335          000020                  $WRDL   =       $TATUS+2        ;WORD COUNT (NOT 2'S COMP)
 5336          000022                  $PACK   =       $WRDL+2         ;WRITE DATA PACK INDICATOR
 5337          000024                  $PREVO  =       $PACK+2         ;PREVIOUS COMMAND SELECTION CODE
 5338          000026                  $PATTC  =       $PREVO+2        ;PATTERN CODE
 5339          000030                  $PREVA  =       $PATTC+2        ;PREVIOUS ADDRESS - TRACK, SECTOR, CYLINDER
 5340          000034                  $OPERC  =       $PREVA+4        ;OPERATION COUNT
 5341          000040                  $POSIT  =       $OPERC+4        ;SEEK COUNT
 5342          000044                  $READ   =       $POSIT+4        ;TOTAL BITS READ COUNT
 5343          000050                  $TOTAL  =       $READ+4         ;TOTAL ERRORS (ALL TYPES) COUNT
 5344          000052                  $SOFT   =       $TOTAL+2        ;'SOFT' ERROR COUNT
 5345          000054                  $HARD   =       $SOFT+2         ;'HARD' ERROR COUNT
 5346          000056                  $SKI    =       $HARD+2         ;'SUSI' ERROR COUNT
 5347          000060                  $MISPO  =       $SKI+2          ;POSITIONING ERROR COUNT
 5348          000062                  $PASSC  =       $MISPO+2        ;PASS COUNTER
 5349          000064                  $FAIR   =       $PASSC+2        ;BUFFER WAIT COUNT
 5350          000066                  $RPDS   =       $FAIR+2         ;RP11E REGISTER STORAGE
 5351          000070                  $RPER   =       $RPDS+2         ;ERROR REGISTER
 5352          000072                  $RPCS           $RPER+2         ;CONTROL REGISTER
 5353          000074                  $RPWC   =       $RPCS+2         ;WORD COUNT REGISTER
 5354          000076                  $RPBA   =       $RPWC+2         ;BUFFER ADDRESS REGISTER
 5355          000100                  $RPCA   =       $RPBA+2         ;CYLINDER ADDRESS REGISTER
 5356          000102                  $RPDA           $RPCA+2         ;SECTOR/ADDRESS REGISTER
 5357          000104                  $RPM1   =       $RPDA+2         ;MAINTENANCE REGISTER #1
 5358          000106                  $SUCA           $RPM1+2         ;SELECTED UNIT CYLINDER ADDR REGISTER
 5359          000110                  $SILO   -       $SUCA+2         ;SILO REGISTER
 5360
 5361
 5362                                  ;BLOCK FOR DRIVE 0
 5363
 5364  024264      000       000       BLK0:   .BYTE   0,0             ;DRIVE NUMBER
 5365  024266  000000                          .WORD   0
 5366  024270  000000                          .WORD   0
 5367  024272  000000                          .WORD   0
 5368  024274  000000                          .WORD   0
 5369  024276  000000                          .WORD   0
 5370  024300  024352                          .WORD   .+52
 5371  024302  000000                          .WORD   0
 5372  024304  000000                          .WORD   0
 5373  024306  000000                          .WORD   0
 5374  024310  000000                          .WORD   0
 5375  024312  000000                          .WORD   0
 5376  024314  000000                          .WORD   0
 5377  024316  000000                          .WORD   0
 5378  024320  000000                          .WORD   0
 5379  024322  000000                          .WORD   0
 5380  024324  000000                          .WORD   0
 5381  024326  000000                          .WORD   0
 5382  024330  000000                          .WORD   0
```

```
 5383  024332  000000                          .WORD   0
 5384  024334  000000                          .WORD   0
 5385  024336  000000                          .WORD   0
 5386  024340  000000                          .WORD   0
 5387  024342  000000                          .WORD   0
 5388  024344  000000                          .WORD   0
 5389  024346  000000                          .WORD   0
 5390  024350  000000                          .WORD   0
 5391  024352  000000                          .WORD   0
 5392  024354  000000                          .WORD   0
 5393  024356  000000                          .WORD   0
 5394  024360  000000                          .WORD   0
 5395  024362  000000                          .WORD   0
 5396  024364  000000                          .WORD   0
 5397  024366  000000                          .WORD   0
 5398  024370  000000                          .WORD   0
 5399  024372  000000                          .WORD   0
 5400  024374  000000                          .WORD   0
 5401
 5402                                  ;BLOCK FOR DRIVE 1
 5403
 5404  024376     001      000        BLK1:    .BYTE   1,0             ;DRIVE NUMBER
 5405  024400  000000                          .WORD   0
 5406  024402  000000                          .WORD   0
 5407  024404  000000                          .WORD   0
 5408  024406  000000                          .WORD   0
 5409  024410  000000                          .WORD   0
 5410  024412  024464                          .WORD   .+52
 5411  024414  000000                          .WORD   0
 5412  024416  000000                          .WORD   0
 5413  024420  000000                          .WORD   0
 5414  024422  000000                          .WORD   0
 5415  024424  000000                          .WORD   0
 5416  024426  000000                          .WORD   0
 5417  024430  000000                          .WORD   0
 5418  024432  000000                          .WORD   0
 5419  024434  000000                          .WORD   0
 5420  024436  000000                          .WORD   0
 5421  024440  000000                          .WORD   0
 5422  024442  000000                          .WORD   0
 5423  024444  000000                          .WORD   0
 5424  024446  000000                          .WORD   0
 5425  024450  000000                          .WORD   0
 5426  024452  000000                          .WORD   0
 5427  024454  000000                          .WORD   0
 5428  024456  000000                          .WORD   0
 5429  024460  000000                          .WORD   0
 5430  024462  000000                          .WORD   0
 5431  024464  000000                          .WORD   0
 5432  024466  000000                          .WORD   0
 5433  024470  000000                          .WORD   0
 5434  024472  000000                          .WORD   0
 5435  024474  000000                          .WORD   0
 5436  024476  000000                          .WORD   0
 5437  024500  000000                          .WORD   0
 5438  024502  000000                          .WORD   0
```

```
5439  024504  000000                          .WORD   0
5440  024506  000000                          .WORD   0
5441
5442                                  ;BLOCK FOR DRIVE 2
5443
5444  024510     002      000         BLK2:   .BYTE   2,0             ;DRIVE NUMBER
5445  024512  000000                          .WORD   0
5446  024514  000000                          .WORD   0
5447  024516  000000                          .WORD   0
5448  024520  000000                          .WORD   0
5449  024522  000000                          .WORD   0
5450  024524  024576                          .WORD   .+52
5451  024526  000000                          .WORD   0
5452  024530  000000                          .WORD   0
5453  024532  000000                          .WORD   0
5454  024534  000000                          .WORD   0
5455  024536  000000                          .WORD   0
5456  024540  000000                          .WORD   0
5457  024542  000000                          .WORD   0
5458  024544  000000                          .WORD   0
5459  024546  000000                          .WORD   0
5460  024550  000000                          .WORD   0
5461  024552  000000                          .WORD   0
5462  024554  000000                          .WORD   0
5463  024556  000000                          .WORD   0
5464  024560  000000                          .WORD   0
5465  024562  000000                          .WORD   0
5466  024564  000000                          .WORD   0
5467  024566  000000                          .WORD   0
5468  024570  000000                          .WORD   0
5469  024572  000000                          .WORD   0
5470  024574  000000                          .WORD   0
5471  024576  000000                          .WORD   0
5472  024600  000000                          .WORD   0
5473  024602  000000                          .WORD   0
5474  024604  000000                          .WORD   0
5475  024606  000000                          .WORD   0
5476  024610  000000                          .WORD   0
5477  024612  000000                          .WORD   0
5478  024614  000000                          .WORD   0
5479  024616  000000                          .WORD   0
5480  024620  000000                          .WORD   0
5481
5482                                  ;BLOCK FOR DRIVE 3
5483
5484  024622     003      000         BLK3:   .BYTE   3,0             ;DRIVE NUMBER
5485  024624  000000                          .WORD   0
5486  024626  000000                          .WORD   0
5487  024630  000000                          .WORD   0
5488  024632  000000                          .WORD   0
5489  024634  000000                          .WORD   0
5490  024636  024710                          .WORD   .+52
5491  024640  000000                          .WORD   0
5492  024642  000000                          .WORD   0
5493  024644  000000                          .WORD   0
5494  024646  000000                          .WORD   0
```

```
 5495  024650  000000                          .WORD   0
 5496  024652  000000                          .WORD   0
 5497  024654  000000                          .WORD   0
 5498  024656  000000                          .WORD   0
 5499  024660  000000                          .WORD   0
 5500  024662  000000                          .WORD   0
 5501  024664  000000                          .WORD   0
 5502  024666  000000                          .WORD   0
 5503  024670  000000                          .WORD   0
 5504  024672  000000                          .WORD   0
 5505  024674  000000                          .WORD   0
 5506  024676  000000                          .WORD   0
 5507  024700  000000                          .WORD   0
 5508  024702  000000                          .WORD   0
 5509  024704  000000                          .WORD   0
 5510  024706  000000                          .WORD   0
 5511  024710  000000                          .WORD   0
 5512  024712  000000                          .WORD   0
 5513  024714  000000                          .WORD   0
 5514  024716  000000                          .WORD   0
 5515  024720  000000                          .WORD   0
 5516  024722  000000                          .WORD   0
 5517  024724  000000                          .WORD   0
 5518  024726  000000                          .WORD   0
 5519  024730  000000                          .WORD   0
 5520  024732  000000                          .WORD   0
 5521
 5522                                  ;BLOCK FOR DRIVE 4
 5523
 5524  024734     004      000        BLK4:    .BYTE   4,0             ;DRIVE NUMBER
 5525  024736  000000                          .WORD   0
 5526  024740  000000                          .WORD   0
 5527  024742  000000                          .WORD   0
 5528  024744  000000                          .WORD   0
 5529  024746  000000                          .WORD   0
 5530  024750  025022                          .WORD   .+52
 5531  024752  000000                          .WORD   0
 5532  024754  000000                          .WORD   0
 5533  024756  000000                          .WORD   0
 5534  024760  000000                          .WORD   0
 5535  024762  000000                          .WORD   0
 5536  024764  000000                          .WORD   0
 5537  024766  000000                          .WORD   0
 5538  024770  000000                          .WORD   0
 5539  024772  000000                          .WORD   0
 5540  024774  000000                          .WORD   0
 5541  024776  000000                          .WORD   0
 5542  025000  000000                          .WORD   0
 5543  025002  000000                          .WORD   0
 5544  025004  000000                          .WORD   0
 5545  025006  000000                          .WORD   0
 5546  025010  000000                          .WORD   0
 5547  025012  000000                          .WORD   0
 5548  025014  000000                          .WORD   0
 5549  025016  000000                          .WORD   0
 5550  025020  000000                          .WORD   0
```

```
5551  025022  000000                          .WORD   0
5552  025024  000000                          .WORD   0
5553  025026  000000                          .WORD   0
5554  025030  000000                          .WORD   0
5555  025032  000000                          .WORD   0
5556  025034  000000                          .WORD   0
5557  025036  000000                          .WORD   0
5558  025040  000000                          .WORD   0
5559  025042  000000                          .WORD   0
5560  025044  000000                          .WORD   0
5561
5562                                  ;BLOCK FOR DRIVE 5
5563
5564  025046     005     000          BLK5:   .BYTE   5,0             ;DRIVE NUMBER
5565  025050  000000                          .WORD   0
5566  025052  000000                          .WORD   0
5567  025054  000000                          .WORD   0
5568  025056  000000                          .WORD   0
5569  025060  000000                          .WORD   0
5570  025062  025134                          .WORD   .+52
5571  025064  000000                          .WORD   0
5572  025066  000000                          .WORD   0
5573  025070  000000                          .WORD   0
5574  025072  000000                          .WORD   0
5575  025074  000000                          .WORD   0
5576  025076  000000                          .WORD   0
5577  025100  000000                          .WORD   0
5578  025102  000000                          .WORD   0
5579  025104  000000                          .WORD   0
5580  025106  000000                          .WORD   0
5581  025110  000000                          .WORD   0
5582  025112  000000                          .WORD   0
5583  025114  000000                          .WORD   0
5584  025116  000000                          .WORD   0
5585  025120  000000                          .WORD   0
5586  025122  000000                          .WORD   0
5587  025124  000000                          .WORD   0
5588  025126  000000                          .WORD   0
5589  025130  000000                          .WORD   0
5590  025132  000000                          .WORD   0
5591  025134  000000                          .WORD   0
5592  025136  000000                          .WORD   0
5593  025140  000000                          .WORD   0
5594  025142  000000                          .WORD   0
5595  025144  000000                          .WORD   0
5596  025146  000000                          .WORD   0
5597  025150  000000                          .WORD   0
5598  025152  000000                          .WORD   0
5599  025154  000000                          .WORD   0
5600  025156  000000                          .WORD   0
5601
5602                                  ;BLOCK FOR DRIVE 6
5603
5604  025160     006     000          BLK6:   .BYTE   6,0             ;DRIVE NUMBER
5605  025162  000000                          .WORD   0
5606  025164  000000                          .WORD   0
```

```
 5607  025166  000000                        .WORD   0
 5608  025170  000000                        .WORD   0
 5609  025172  000000                        .WORD   0
 5610  025174  025246                        .WORD   .+52
 5611  025176  000000                        .WORD   0
 5612  025200  000000                        .WORD   0
 5613  025202  000000                        .WORD   0
 5614  025204  000000                        .WORD   0
 5615  025206  000000                        .WORD   0
 5616  025210  000000                        .WORD   0
 5617  025212  000000                        .WORD   0
 5618  025214  000000                        .WORD   0
 5619  025216  000000                        .WORD   0
 5620  025220  000000                        .WORD   0
 5621  025222  000000                        .WORD   0
 5622  025224  000000                        .WORD   0
 5623  025226  000000                        .WORD   0
 5624  025230  000000                        .WORD   0
 5625  025232  000000                        .WORD   0
 5626  025234  000000                        .WORD   0
 5627  025236  000000                        .WORD   0
 5628  025240  000000                        .WORD   0
 5629  025242  000000                        .WORD   0
 5630  025244  000000                        .WORD   0
 5631  025246  000000                        .WORD   0
 5632  025250  000000                        .WORD   0
 5633  025252  000000                        .WORD   0
 5634  025254  000000                        .WORD   0
 5635  025256  000000                        .WORD   0
 5636  025260  000000                        .WORD   0
 5637  025262  000000                        .WORD   0
 5638  025264  000000                        .WORD   0
 5639  025266  000000                        .WORD   0
 5640  025270  000000                        .WORD   0
 5641
 5642                              ;BLOCK FOR DRIVE 7
 5643
 5644  025272    007     000     BLK7:       .BYTE   7,0             ;DRIVE NUMBER
 5645  025274  000000                        .WORD   0
 5646  025276  000000                        .WORD   0
 5647  025300  000000                        .WORD   0
 5648  025302  000000                        .WORD   0
 5649  025304  000000                        .WORD   0
 5650  025306  025360                        .WORD   .+52
 5651  025310  000000                        .WORD   0
 5652  025312  000000                        .WORD   0
 5653  025314  000000                        .WORD   0
 5654  025316  000000                        .WORD   0
 5655  025320  000000                        .WORD   0
 5656  025322  000000                        .WORD   0
 5657  025324  000000                        .WORD   0
 5658  025326  000000                        .WORD   0
 5659  025330  000000                        .WORD   0
 5660  025332  000000                        .WORD   0
 5661  025334  000000                        .WORD   0
 5662  025336  000000                        .WORD   0
```

```
 5663  025340  000000                          .WORD   0
 5664  025342  000000                          .WORD   0
 5665  025344  000000                          .WORD   0
 5666  025346  000000                          .WORD   0
 5667  025350  000000                          .WORD   0
 5668  025352  000000                          .WORD   0
 5669  025354  000000                          .WORD   0
 5670  025356  000000                          .WORD   0
 5671  025360  000000                          .WORD   0
 5672  025362  000000                          .WORD   0
 5673  025364  000000                          .WOR.   0
 5674  025366  000000                          .WORL   0
 5675  025370  000000                          .WORD   0
 5676  025372  000000                          .WORD   0
 5677  025374  000000                          .WORD   0
 5678  025376  000000                          .WORD   0
 5679  025400  000000                          .WORD   0
 5680  025402  000000                          .WORD   0
 5681
 5682                                  ;GENERAL PURPOSE DPB - USED BY 'READHD' & 'RECALT'
 5683
 5684  025404  000000  000000  177775  GENDPB: .WORD   0,0,-3,CYLDER
 5685  025412  033626
 5686  025414  000000  000000  025424          .WORD   0,0,GENREG,0
 5687  025422  000000
 5688  025424  000012                  GENREG: .BLKW   12              ;REGISTER STORAGE IF ERROR
 5689
 5690                                  ;;*****************************************************************
 5691
 5692                                  .SBTTL  TABLES, CONSTANTS, AND VARIABLE LOCATIONS
 5693
 5694                                  ;;*****************************************************************
 5695
 5696  025450  000000                  ASNLST: .WORD   0               ;BIT SET FOR EACH ASSIGNED DRIVE
 5697
 5698  025452  000000  000000  000000  ORDERQ: .WORD   0,0,0,0,0,0,0,0  ;LIST OF DRIVES PERFORMING COMMANDS
 5699  025460  000000  000000  000000
 5700  025466  000000  000000  000000
 5701
 5702  025474  000000  000000  000000  DUNIT:  .WORD   0,0,0,0,0,0,0,0  ;ADDRESSES OF DRIVES TO BE DEASSIGNED
 5703  025502  000000  000000  000000
 5704  025510  000000  000000  000000
 5705
 5706  025516  000000  000000  000000  NEWUNT: .WORD   0,0,0,0,0,0,0,0  ;ADDRESSES OF NEWLY ASSIGNED DRIVES
 5707  025524  000000  000000  000000
 5708  025532  000000  000000  000000
 5709
 5710  025540  000000  000000  000000  AVAIL:  .WORD   0,0,0,0,0,0,0,0  ;LIST OF DRIVES WAITING FOR PARAMETERS
 5711  025546  000000  000000  000000
 5712  025554  000000  000000  000000
 5713
 5714  025562  000000  000000  000000  WAIT:   .WORD   0,0,0,0,0,0,0,0  ;LIST OF DRIVES WAITING FOR BUFFERS
 5715  025570  000000  000000  000000
 5716  025576  000000  000000  000000
 5717
 5718  025604  000000                  BUFTBL: .WORD   0               ;BUFFER ALLOCATION TABLE ENTRY COUNT
```

```
 5719  025606  000000  000000                  .WORD   0,0
 5720  025612  000000  000000                  .WORD   0,0
 5721  025616  000000  000000                  .WORD   0,0
 5722  025622  000000  000000                  .WORD   0,0
 5723  025626  000000  000000                  .WORD   0,0
 5724  025632  000000  000000                  .WORD   0,0
 5725  025636  000000  000000                  .WORD   0,0
 5726  025642  000000  000000                  .WORD   0,0
 5727  025646  000000  000000                  .WORD   0,0
 5728  025652  000000  000000                  .WORD   0,0
 5729  025656  000000  000000                  .WORD   0,0
 5730  025662  000000  000000                  .WORD   0,0
 5731  025666  000000  000000                  .WORD   0,0
 5732  025672  000000  000000                  .WORD   0,0
 5733  025676  000000  000000                  .WORD   0,0
 5734  025702  000000  000000                  .WORD   0,0
 5735  025706  000000  000000                  .WORD   0,0
 5736  025712  000000  000000                  .WORD   0,0
 5737  025716  000000  000000                  .WORD   0,0
 5738  025722  000000  000000                  .WORD   0,0
 5739
 5740  025726                          BLKADR:                 ;ADDRESS OF EACH UNIT BLOCK START
 5741  025726  024264                          .WORD   BLK0
 5742  025730  024376                          .WORD   BLK1
 5743  025732  024510                          .WORD   BLK2
 5744  025734  024622                          .WORD   BLK3
 5745  025736  024734                          .WORD   BLK4
 5746  025740  025046                          .WORD   BLK5
 5747  025742  025160                          .WORD   BLK6
 5748  025744  025272                          .WORD   BLK7
 5749
 5750  025746     003                  COMTBL: .BYTE   WRTSEK          ;WRITE WITH IMPLIED SEEK (CODE = 0)
 5751  025747     013                          .BYTE   WRITE           ;WRITE WITHOUT IMPLIED SEEK (CODE = 1)
 5752  025750     007                          .BYTE   WRTCHK          ;WRITE CHECK (CODE = 2)
 5753  025751     005                          .BYTE   RDSEK           ;READ WITH IMPLIED SEEK (CODE = 3)
 5754  025752     017                          .BYTE   READ            ;READ WITHOUT IMPLIED SEEK (CODE = 4)
 5755
 5756
 5757          025754                          .EVEN
 5758
 5759                                  ;;******************************************************************
 5760
 5761                                  .SBTTL  DATA PATTERNS
 5762
 5763                                  ;;******************************************************************
 5764
 5765  025754  000000                  STNDAT: .WORD   0               ;STANDARD DATA PATTERN POINTER TABLE
 5766  025756  026060                          .WORD   DATA1
 5767  025760  026120                          .WORD   DATA1+40
 5768  025762  026160                          .WORD   DATA1+100
 5769  025764  026220                          .WORD   DATA1+140
 5770  025766  026260                          .WOPD   DATA1+200
 5771  025770  026320                          .WORD   DATA1+240
 5772  025772  026360                          .WORD   DATA1+300
 5773  025774  026420                          .WORD   DATA1+340
 5774  025776  026460                          .WORD   DATA1+400
```

```
 5775  026000  026520                          .WORD   DATA1+440
 5776  026002  026560                          .WORD   DATA1+500
 5777  026004  026620                          .WORD   DATA1+540
 5778  026006  026660                          .WORD   DATA1+600
 5779  026010  026720                          .WORD   DATA1+640
 5780  026012  026760                          .WORD   DATA1+700
 5781  026014  026020                          .WORD   DATA0           ;ZEROES
 5782  026016  026722                          .WORD   DATA1+642       ;ONES
 5783
 5784  026020  000000                  DATA0:  .WORD   0       ;STANDARD PATTERN 0
 5785  026022  000000                          .WORD   0
 5786  026024  000000                          .WORD   0
 5787  026026  000000                          .WORD   0
 5788  026030  000000                          .WORD   0
 5789  026032  000000                          .WORD   0
 5790  026034  000000                          .WORD   0
 5791  026036  000000                          .WORD   0
 5792  026040  000000                          .WORD   0
 5793  026042  000000                          .WORD   0
 5794  026044  000000                          .WORD   0
 5795  026046  000000                          .WORD   0
 5796  026050  000000                          .WORD   0
 5797  026052  000000                          .WORD   0
 5798  026054  000000                          .WORD   0
 5799  026056  000000                          .WORD   0
 5800
 5801  026060  000001                  DATA1:  .WORD   000001  ;STANDARD PATTERN 1
 5802  026062  000003                          .WORD   000003
 5803  026064  000007                          .WORD   000007
 5804  026066  000017                          .WORD   000017
 5805  026070  000037                          .WORD   000037
 5806  026072  000077                          .WORD   000077
 5807  026074  000177                          .WORD   000177
 5808  026076  000377                          .WORD   000377
 5809  026100  000777                          .WORD   000777
 5810  026102  001777                          .WORD   001777
 5811  026104  003777                          .WORD   003777
 5812  026106  007777                          .WORD   007777
 5813  026110  017777                          .WORD   017777
 5814  026112  037777                          .WORD   037777
 5815  026114  077777                          .WORD   077777
 5816  026116  177777                          .WORD   177777
 5817
 5818  026120  177776                          .WORD   177776  ;STANDARD PATTERN 2
 5819  026122  177774                          .WORD   177774
 5820  026124  177770                          .WORD   177770
 5821  026126  177760                          .WORD   177760
 5822  026130  177740                          .WORD   177740
 5823  026132  177700                          .WORD   177700
 5824  026134  177600                          .WORD   177600
 5825  026136  177400                          .WORD   177400
 5826  026140  177000                          .WORD   177000
 5827  026142  176000                          .WORD   176000
 5828  026144  174000                          .WORD   174000
 5829  026146  170000                          .WORD   170000
 5830  026150  160000                          .WORD   160000
```

```
 5831  026152  140000                          .WORD   140000
 5832  026154  100000                          .WORD   100000
 5833  026156  000000                          .WORD   000000
 5834
 5835  026160  000000                          .WORD   000000  ;STANDARD PATTERN 3
 5836  026162  000000                          .WORD   000000
 5837  026164  000000                          .WORD   000000
 5838  026166  177777                          .WORD   177777
 5839  026170  177777                          .WORD   177777
 5840  026172  177777                          .WORD   177777
 5841  026174  000000                          .WORD   000000
 5842  026176  000000                          .WORD   000000
 5843  026200  177777                          .WORD   177777
 5844  026202  177777                          .WORD   177777
 5845  026204  000000                          .WORD   000000
 5846  026206  177777                          .WORD   177777
 5847  026210  000000                          .WORD   000000
 5848  026212  177777                          .WORD   177777
 5849  026214  000000                          .WORD   000000
 5850  026216  177777                          .WORD   177777
 5851
 5852  026220  000000                          .WORD   000000  ;STANDARD PATTERN 4
 5853  026222  010421                          .WORD   010421
 5854  026224  021042                          .WORD   021042
 5855  026226  031463                          .WORD   031463
 5856  026230  042104                          .WORD   042104
 5857  026232  052525                          .WORD   052525
 5858  026234  063146                          .WORD   063146
 5859  026236  073567                          .WORD   073567
 5860  026240  104210                          .WORD   104210
 5861  026242  114631                          .WORD   114631
 5862  026244  125252                          .WORD   125252
 5863  026246  135673                          .WORD   135673
 5864  026250  146314                          .WORD   146314
 5865  026252  156735                          .WORD   156735
 5866  026254  167356                          .WORD   167356
 5867  026256  177777                          .WORD   177777
 5868
 5869  026260  052525                          .WORD   052525  ;STANDARD PATTERN 5
 5870  026262  052525                          .WORD   052525
 5871  026264  052525                          .WORD   052525
 5872  026266  125252                          .WORD   125252
 5873  026270  125252                          .WORD   125252
 5874  026272  125252                          .WORD   125252
 5875  026274  052525                          .WORD   052525
 5876  026276  052525                          .WORD   052525
 5877  026300  125252                          .WORD   125252
 5878  026302  125252                          .WORD   125252
 5879  026304  052525                          .WORD   052525
 5880  026306  125252                          .WORD   125252
 5881  026310  052525                          .WORD   052525
 5882  026312  125252                          .WORD   125252
 5883  026314  052525                          .WORD   052525
 5884  026316  125252                          .WORD   125252
 5885
 5886  026320  007417                          .WORD   007417  ;STANDARD PATTERN 6
```

```
 5887  026322  007417                          .WORD   007417
 5888  026324  007417                          .WORD   007417
 5889  026326  170360                          .WORD   170360
 5890  026330  170360                          .WORD   170360
 5891  026332  170360                          .WORD   170360
 5892  026334  007417                          .WORD   007417
 5893  026336  007417                          .WORD   007417
 5894  026340  170360                          .WORD   170360
 5895  026342  170360                          .WORD   170360
 5896  026344  007417                          .WORD   007417
 5897  026346  170360                          .WORD   170360
 5898  026350  007417                          .WORD   007417
 5899  026352  170360                          .WORD   170360
 5900  026354  007417                          .WORD   007417
 5901  026356  170360                          .WORD   170360
 5902
 5903  026360  026455                          .WORD   026455  ;STANDARD PATTERN 7
 5904  026362  026455                          .WORD   026455
 5905  026364  026455                          .WORD   026455
 5906  026366  151322                          .WORD   151322
 5907  026370  151322                          .WORD   151322
 5908  026372  151322                          .WORD   151322
 5909  026374  026455                          .WORD   026455
 5910  026376  026455                          .WORD   026455
 5911  026400  151322                          .WORD   151322
 5912  026402  151322                          .WORD   151322
 5913  026404  026455                          .WORD   026455
 5914  026406  151322                          .WORD   151322
 5915  026410  026455                          .WORD   026455
 5916  026412  151322                          .WORD   151322
 5917  026414  026455                          .WORD   026455
 5918  026416  151322                          .WORD   151322
 5919
 5920  026420  077577                          .WORD   077577  ;STANDARD PATTERN 8 (WORST CASE PATTERN)
 5921  026422  077577                          .WORD   077577
 5922  026424  077577                          .WORD   077577
 5923  026426  077577                          .WORD   077577
 5924  026430  077577                          .WORD   077577
 5925  026432  077577                          .WORD   077577
 5926  026434  077577                          .WORD   077577
 5927  026436  077577                          .WORD   077577
 5928  026440  077577                          .WORD   077577
 5929  026442  077577                          .WORD   077577
 5930  026444  077577                          .WORD   077577
 5931  026446  077577                          .WORD   077577
 5932  026450  077577                          .WORD   077577
 5933  026452  077577                          .WORD   077577
 5934  026454  077577                          .WORD   077577
 5935  026456  077577                          .WORD   077577
 5936
 5937  026460  000001                          .WORD   000001  ;STANDARD PATTERN 9
 5938  026462  000002                          .WORD   000002
 5939  026464  000004                          .WORD   000004
 5940  026466  000010                          .WORD   000010
 5941  026470  000020                          .WORD   000020
 5942  026472  000040                          .WORD   000040
```

```
 5943  026474  000100                          .WORD   000100
 5944  026476  000200                          .WORD   000200
 5945  026500  000400                          .WORD   000400
 5946  026502  001000                          .WORD   001000
 5947  026504  002000                          .WORD   002000
 5948  026506  004000                          .WORD   004000
 5949  026510  010000                          .WORD   010000
 5950  026512  020000                          .WORD   020000
 5951  026514  040000                          .WORD   040000
 5952  026516  100000                          .WORD   100000
 5953
 5954  026520  177776                          .WORD   177776  ;STANDARD PATTERN 10
 5955  026522  177775                          .WORD   177775
 5956  026524  177773                          .WORD   177773
 5957  026526  177767                          .WORD   177767
 5958  026530  177757                          .WORD   177757
 5959  026532  177737                          .WORD   177737
 5960  026534  177677                          .WORD   177677
 5961  026536  177577                          .WORD   177577
 5962  026540  177377                          .WORD   177377
 5963  026542  176777                          .WORD   176777
 5964  026544  175777                          .WORD   175777
 5965  026546  173777                          .WORD   173777
 5966  026550  167777                          .WORD   167777
 5967  026552  157777                          .WORD   157777
 5968  026554  137777                          .WORD   137777
 5969  026556  077777                          .WORD   077777
 5970
 5971  026560  172666                          .WORD   172666  ;STANDARD PATTERN 11
 5972  026562  155555                          .WORD   155555
 5973  026564  172666                          .WORD   172666
 5974  026566  155555                          .WORD   155555
 5975  026570  172666                          .WORD   172666
 5976  026572  155555                          .WORD   155555
 5977  026574  172666                          .WORD   172666
 5978  026576  155555                          .WORD   155555
 5979  026600  172666                          .WORD   172666
 5980  026602  155555                          .WORD   155555
 5981  026604  172666                          .WORD   172666
 5982  026606  155555                          .WORD   155555
 5983  026610  172666                          .WORD   172666
 5984  026612  155555                          .WORD   155555
 5985  026614  172666                          .WORD   172666
 5986  026616  155555                          .WORD   155555
 5987
 5988  026620  077777                          .WORD   077777  ;STANDARD PATTERN 12
 5989  026622  137777                          .WORD   137777
 5990  026624  157777                          .WORD   157777
 5991  026626  167777                          .WORD   167777
 5992  026630  173777                          .WORD   173777
 5993  026632  175777                          .WORD   175777
 5994  026634  176777                          .WORD   176777
 5995  026636  177377                          .WORD   177377
 5996  026640  177577                          .WORD   177577
 5997  026642  177677                          .WORD   177677
 5998  026644  177737                          .WORD   177737
```

```
 5999  026646  177757                          .WORD   177757
 6000  026650  177767                          .WORD   177767
 6001  026652  177773                          .WORD   177773
 6002  026654  177775                          .WORD   177775
 6003  026656  177776                          .WORD   177776
 6004
 6005  026660  153333                          .WORD   153333  ;STANDARD PATTERN 13
 6006  026662  066667                          .WORD   066667
 6007  026664  153333                          .WORD   153333
 6008  026666  066667                          .WORD   066667
 6009  026670  153333                          .WORD   153333
 6010  026672  066667                          .WORD   066667
 6011  026674  153333                          .WORD   153333
 6012  026676  066667                          .WORD   066667
 6013  026700  153333                          .WORD   153333
 6014  026702  066667                          .WORD   066667
 6015  026704  153333                          .WORD   153333
 6016  026706  066667                          .WORD   066667
 6017  026710  153333                          .WORD   153333
 6018  026712  066667                          .WORD   066667
 6019  026714  153333                          .WORD   153333
 6020  026716  066667                          .WORD   066667
 6021
 6022  026720  000000                          .WORD   000000  ;STANDARD PATTERN 14
 6023  026722  000000                          .WORD   000000
 6024  026724  177777                          .WORD   177777
 6025  026726  177777                          .WORD   177777
 6026  026730  177777                          .WORD   177777
 6027  026732  177777                          .WORD   177777
 6028  026734  177777                          .WORD   177777
 6029  026736  177777                          .WORD   177777
 6030  026740  177777                          .WORD   177777
 6031  026742  177777                          .WORD   177777
 6032  026744  177777                          .WORD   177777
 6033  026746  177777                          .WORD   177777
 6034  026750  177777                          .WORD   177777
 6035  026752  177777                          .WORD   177777
 6036  026754  177777                          .WORD   177777
 6037  026756  177777                          .WORD   177777
 6038
 6039  026760  177777                          .WORD   177777  ;STANDARD PATTERN 15
 6040  026762  177777                          .WORD   177777
 6041  026764  000000                          .WORD   000000
 6042  026766  000000                          .WORD   000000
 6043  026770  000000                          .WORD   000000
 6044  026772  000000                          .WORD   000000
 6045  026774  000000                          .WORD   000000
 6046  026776  000000                          .WORD   000000
 6047  027000  000000                          .WORD   000000
 6048  027002  000000                          .WORD   000000
 6049  027004  000000                          .WORD   000000
 6050  027006  000000                          .WORD   000000
 6051  027010  000000                          .WORD   000000
 6052  027012  000000                          .WORD   000000
 6053  027014  000000                          .WORD   000000
 6054  027016  000000                          .WORD   000000
```

```
 6055
 6056                                   ;;*****************************************************************
 6057
 6058                                   .SBTTL  PRINTER/TELETYPE MESSAGES
 6059
 6060                                   ;;*****************************************************************
 6061
 6062  027020  050122  030461  042040  EM1:    .ASCIZ  @RP11 DIDN'T RESPOND TO ADDRESSING@
 6063  027026  042111  023516  020124
 6064  027034  042522  050123  047117
 6065  027042  020104  047524  040440
 6066  027050  042104  042522  051523
 6067  027056  047111  000107
 6068
 6069  027062  047111  040526  044514  EM2:    .ASCIZ  @INVALID RP11 COMMAND@
 6070  027070  020104  050122  030461
 6071  027076  041440  046517  040515
 6072  027104  042116     000
 6073
 6074  027107     125  042116  043105  EM3:    .ASCIZ  @UNDEFINED INTERRUPT FROM THE RP11@
 6075  027114  047111  042105  044440
 6076  027122  052116  051105  052522
 6077  027130  052120  043040  047522
 6078  027136  020115  044124  020105
 6079  027144  050122  030461     000
 6080
 6081  027151     103  047101  052047  EM10:   .ASCIZ  @CAN'T MATCH DATA READ WITH A PATTERN@
 6082  027156  046440  052101  044103
 6083  027164  042040  052101  020101
 6084  027172  042522  042101  053440
 6085  027200  052111  020110  020101
 6086  027206  040520  052124  051105
 6087  027214  000116
 6088
 6089  027216  040504  040524  041440  EM12:   .ASCIZ  @DATA COMPARE ERRORS@
 6090  027224  046517  040520  042522
 6091  027232  042440  051122  051117
 6092  027240  000123
 6093
 6094  027242  042447  051122  020047  EM15:   .ASCIZ  @'ERR' NOT SET BUT 'RPER' CONTENTS NOT ZERO@
 6095  027250  047516  020124  042523
 6096  027256  020124  052502  020124
 6097  027264  051047  042520  023522
 6098  027272  041440  047117  042524
 6099  027300  052116  020123  047516
 6100  027306  020124  042532  047522
 6101  027314     000
 6102
 6103  027315     127  051117  020104  EM16:   .ASCIZ  @WORD COUNT NOT ZERO OR BUFFER ADDRESS WRONG@
 6104  027322  047503  047125  020124
 6105  027330  047516  020124  042532
 6106  027336  047522  047440  020122
 6107  027344  052502  043106  051105
 6108  027352  040440  042104  042522
 6109  027360  051523  053440  047522
 6110  027366  043516     000
```

```
 6111
 6112  027371     123  043117  053524  EM17:   .ASCIZ  @SOFTWARE TIMEOUT DURING POSITIONING@
 6113  027376  051101  020105  044524
 6114  027404  042515  052517  020124
 6115  027412  052504  044522  043516
 6116  027420  050040  051517  052111
 6117  027426  047511  044516  043516
 6118  027434     000
 6119
 6120  027435     123  043117  053524  EM20:   .ASCIZ  @SOFTWARE TIMEOUT DURING I/O@
 6121  027442  051101  020105  044524
 6122  027450  042515  052517  020124
 6123  027456  052504  044522  043516
 6124  027464  044440  047457     000
 6125
 6126  027471     104  044522  042526  EM21:   .ASCIZ  @DRIVE OFFLINE@
 6127  027476  047440  043106  044514
 6128  027504  042516     000
 6129
 6130  027507     104  044522  042526  EM22:   .ASCIZ  @DRIVE UNSAFE@
 6131  027514  052440  051516  043107
 6132  027522  000105
 6133
 6134  027524  042523  045505  044440  EM23:   .ASCIZ  @SEEK INCOMPLETE@
 6135  027532  041516  046517  046120
 6136  027540  052105  000105
 6137
 6138  027544  047125  041111  051525  EM24:   .ASCIZ  @UNIBUS TRANSFER ERROR@
 6139  027552  052040  040522  051516
 6140  027560  042506  020122  051105
 6141  027566  047522  000122
 6142
 6143  027572  047503  052116  047522  EM25:   .ASCIZ  @CONTROLLER ERROR@
 6144  027600  046114  051105  042440
 6145  027606  051122  051117     000
 6146
 6147  027613     116  047117  042455  EM26:   .ASCIZ  @NON-EXISTENT DISK ADDRESS@
 6148  027620  044530  052123  047105
 6149  027626  020124  044504  045523
 6150  027634  040440  042104  042522
 6151  027642  051523     000
 6152
 6153  027645     104  052101  020101  EM27:   .ASCIZ  @DATA ERROR@
 6154  027652  051105  047522  000122
 6155
 6156  027660  051127  052111  020105  EM32:   .ASCIZ  @WRITE CHECK ERROR (NO DATA ERROR)@
 6157  027666  044103  041505  020113
 6158  027674  051105  047522  020122
 6159  027702  047050  020117  040504
 6160  027710  040524  042440  051122
 6161  027716  051117  000051
 6162
 6163  027722  051127  052111  020105  EM33:   .ASCIZ  @WRITE CHECK ERROR (DATA ERROR BITS SET)@
 6164  027730  044103  041505  020113
 6165  027736  051105  047522  020122
 6166  027744  042050  052101  020101
```

```
 6167  027752  051105  047522  020122
 6168  027760  044502  051524  051440
 6169  027766  052105  000051
 6170
 6171  027772  042510  042101  051105  EM34:   .ASCIZ  @HEADER NOT FOUND@
 6172  030000  047040  052117  043040
 6173  030006  052517  042116     000
 6174
 6175  030013     106  051117  040515  EM35:   .ASCIZ  @FORMAT ERROR WHILE SEARCHING FOR INITIAL SECTOR@
 6176  030020  020124  051105  047522
 6177  030026  020122  044127  046111
 6178  030034  020105  042523  051101
 6179  030042  044103  047111  020107
 6180  030050  047506  020122  047111
 6181  030056  052111  040511  020114
 6182  030064  042523  052103  051117
 6183  030072     000
 6184
 6185  030073     110  040505  042504  EM36:   .ASCIZ  @HEADER NOT FOUND, CYLINDER IN HEADER DOES NOT COMPARE@
 6186  030100  020122  047516  020124
 6187  030106  047506  047125  026104
 6188  030114  041440  046131  047111
 6189  030122  042504  020122  047111
 6190  030130  044040  040505  042504
 6191  030136  020122  047504  051505
 6192  030144  047040  052117  041440
 6193  030152  046517  040520  042522
 6194  030160     000
 6195
 6196  030161     104  043111  042506  EM37:   .ASCIZ  @DIFFERENT ERROR DURING RETRY@
 6197  030166  042522  052116  042440
 6198  030174  051122  051117  042040
 6199  030202  051125  047111  020107
 6200  030210  042522  051124  000131
 6201
 6202  030216  042447  051122  020047  EM43:   .ASCIZ  @'ERR' SET BUT NOT ERROR BITS SET@
 6203  030224  042523  020124  052502
 6204  030232  020124  047516  020124
 6205  030240  051105  047522  020122
 6206  030246  044502  051524  051440
 6207  030254  052105     000
 6208
 6209  030257     127  044522  042524  EM44:   .ASCIZ  @WRITE PROTECTION VIOLATION@
 6210  030264  050040  047522  042524
 6211  030272  052103  047511  020116
 6212  030300  044526  046117  052101
 6213  030306  047511  000116
 6214
 6215  030312  047506  046522  052101  EM45:   .ASCIZ  @FORMAT ERROR DURING I/O@
 6216  030320  042440  051122  051117
 6217  030326  042040  051125  047111
 6218  030334  020107  027511  000117
 6219
 6220  030342  050122  042101  000122  DH1:    .ASCIZ  @RPADR@
 6221  030350  051104  020126  020043  DH2:    .ASCIZ  @DRV #   RPCS@
 6222  030356  020040  050122  051503
```

```
 6223  030364      000
 6224  030365      122  042120  020123  DH3:    .ASCIZ  @RPDS    RPCS@
 6225  030372   020040  051040  041520
 6226  030400   000123
 6227  030402   020040  020040  020040  DH10:   .ASCII  @        BUFFER@<15><12>
 6228  030410   020040  052502  043106
 6229  030416   051105  005015
 6230  030422   051104  020126  020043          .ASCIZ  @DRV #   START   CYL     TRK     SEC@
 6231  030430   020040  052123  051101
 6232  030436   020124  020040  054503
 6233  030444   020114  020040  020040
 6234  030452   051124  020113  020040
 6235  030460   020040  042523  000103
 6236  030466   042101  051104  020040  DH10A:  .ASCIZ  @ADDR    CONTENTS@
 6237  030474   020040  047503  052116
 6238  030502   047105  051524     000
 6239  030507      040  020040  020040  DH12:   .ASCII  @        GOOD    BAD@<15><12>
 6240  030514   020040  043440  047517
 6241  030522   020104  020040  041040
 6242  030530   042101  005015
 6243  030534   042101  051104  020040          .ASCIZ  @ADDR    DATA    DATA@
 6244  030542   020040  040504  040524
 6245  030550   020040  020040  040504
 6246  030556   040524     000
 6247  030561      124  052117  046101  DH14:   .ASCIZ  @TOTAL COMPARE ERRORS:@
 6248  030566   041440  046517  040520
 6249  030574   042522  042440  051122
 6250  030602   051117  035123     000
 6251  030607      104  053122  021440  DH15:   .ASCIZ  @DRV #   RPDS    RPER    RPCS@
 6252  030614   020040  051040  042120
 6253  030622   020123  020040  051040
 6254  030630   042520  020122  020040
 6255  030636   051040  041520  000123
 6256  030644   020040  020040  020040  DH16:   .ASCII  @        EXPTD   ACTUAL@<15><12>
 6257  030652   020040  054105  052120
 6258  030660   020104  020040  041501
 6259  030666   052524  046101  005015
 6260  030674   051104  020126  020043          .ASCIZ  @DRV #   RPBA    RPBA    RPWC    RPDS    RPER    RPCS    @
 6261  030702   020040  050122  040502
 6262  030710   020040  020040  050122
 6263  030716   040502  020040  020040
 6264  030724   050122  041527  020040
 6265  030732   020040  050122  051504
 6266  030740   020040  020040  050122
 6267  030746   051105  020040  020040
 6268  030754   050122  051503  020040
 6269  030762   020040     000
 6270  030765      104  053122  021440  DH17:   .ASCIZ  @DRV #   RPDS    RPER    RPCS    RPWC    RPBA@
 6271  030772   020040  051040  042120
 6272  031000   020123  020040  051040
 6273  031006   042520  020122  020040
 6274  031014   051040  041520  020123
 6275  031022   020040  051040  053520
 6276  031030   020103  020040  051040
 6277  031036   041120  000101
 6278  031042   050122  040503  020040  DH17A:  .ASCIZ  @RPCA    RPDA    RPM1    SUCA    SILO@
```

```
 6279  031050  020040  050122  040504
 6280  031056  020040  020040  050122
 6281  031064  030515  020040  020040
 6282  031072  052523  040503  020040
 6283  031100  020040  044523  047514
 6284  031106     000
 6285  031107     040  020040  020040  DH23:   .ASCII  @       PREV    DEST@<15><12>
 6286  031114  020040  050040  042522
 6287  031122  020126  020040  042040
 6288  031130  051505  006524     012
 6289  031135     104  053122  021440          .ASCIZ  @DRV #  CYL     CYL     RPDS    RPER    RPCS@
 6290  031142  020040  041440  046131
 6291  031150  020040  020040  041440
 6292  031156  046131  020040  020040
 6293  031164  051040  042120  020123
 6294  031172  020040  051040  042520
 6295  031200  020122  020040  051040
 6296  031206  041520  000123
 6297  031212  050122  040503  020040  DH23A:  .ASCIZ  @RPCA    SUCA@
 6298  031220  020040  052523  040503
 6299  031226     000
 6300  031227     104  053122  021440  DH24:   .ASCIZ  @DRV #  RPDS    RPER    RPCS    RPBA@
 6301  031234  020040  051040  042120
 6302  031242  020123  020040  051040
 6303  031250  042520  020122  020040
 6304  031256  051040  041520  020123
 6305  031264  020040  051040  041120
 6306  031272  000101
 6307  031274  051104  020126  020043  DH25:   .ASCIZ  @DRV #  RPDS    RPER    RPCS@
 6308  031302  020040  050122  051504
 6309  031310  020040  020040  050122
 6310  031316  051105  020040  020040
 6311  031324  050122  051503     000
 6312  031331     104  053122  021440  DH26:   .ASCIZ  @DRV #  RPDS    RPER    RPCS    RPCA    RPDA@
 6313  031336  020040  051040  042120
 6314  031344  020123  020040  051040
 6315  031352  042520  020122  020040
 6316  031360  051040  041520  020123
 6317  031366  020040  051040  041520
 6318  031374  020101  020040  051040
 6319  031402  042120  000101
 6320  031406  051104  020126  020043  DH27:   .ASCIZ  @DRV #  CYL     TRK     SEC     RPDS    RPER    RPCS@
 6321  031414  020040  054503  020114
 6322  031422  020040  020040  051124
 6323  031430  020113  020040  020040
 6324  031436  042523  020103  020040
 6325  031444  020040  050122  051504
 6326  031452  020040  020040  050122
 6327  031460  051105  020040  020040
 6328  031466  050122  051503     000
 6329  031473     122  053520  020103  DH27A:  .ASCIZ  @RPWC    RPBA    RPCA    RPDA    RPM1    SUCA    SILO@
 6330  031500  020040  051040  041120
 6331  031506  020101  020040  051040
 6332  031514  041520  020101  020040
 6333  031522  051040  042120  020101
 6334  031530  020040  051040  046520
```

```
 6335  031536  020061  020040  051440
 6336  031544  041525  020101  020040
 6337  031552  051440  046111  000117
 6338  031560  047523  052106  042440  DH30:   .ASCIZ  @SOFT ERROR@
 6339  031566  051122  051117     000
 6340  031573     110  051101  020104  DH31:   .ASCIZ  @HARD ERROR@
 6341  031600  051105  047522  000122
 6342  031606  051104  020126  020043  DH34:   .ASCIZ  @DRV # PREV CYL  RPDS    RPER    RPCS    RPDA@
 6343  031614  051120  053105  041440
 6344  031622  046131  020040  050122
 6345  031630  051504  020040  020040
 6346  031636  050122  051105  020040
 6347  031644  020040  050122  051503
 6348  031652  020040  020040  050122
 6349  031660  040504     000
 6350  031663     122  041520  020101  DH34A:  .ASCIZ  @RPCA    SUCA    HEADER (AS READ INTO MEM)@
 6351  031670  020040  051440  041525
 6352  031676  020101  020040  044040
 6353  031704  040505  042504  020122
 6354  031712  040450  020123  042522
 6355  031720  042101  044440  052116
 6356  031726  020117  042515  024515
 6357  031734     000
 6358  031735     122  052105  054522  DH41:   .ASCIZ  @RETRY SUCESSFUL@
 6359  031742  051440  041525  051505
 6360  031750  043123  046125     000
 6361  031755     122  052105  054522  DH42:   .ASCIZ  @RETRY UNSUCESSFUL@
 6362  031762  052440  051516  041525
 6363  031770  051505  043123  046125
 6364  031776     000
 6365
 6366          032000                          .EVEN
 6367
 6368  032000  001216                  DT1:    .WORD   RPADR
 6369  032002  001264  001166          DT2:    .WORD   DRIVE,$REG0+RPCS
 6370  032006  021074  021076          DT3:    .WORD   RPERRS,RPERRS+2
 6371  032012  001122  001126          DT11:   .WORD   $BDADR,$BDDAT
 6372  032016  001264  013154  013146  DT10:   .WORD   DRIVE,CMBUF,CMCYL,CMTRK,CMSEC
 6373  032024  013152  013150
 6374  032030  001122  001124  001126  DT13:   .WORD   $BDADR,$GDDAT,$BDDAT
 6375  032036  013140                  DT14:   .WORD   ERCTR
 6376  032040  001264  001162  001164  DT15:   .WORD   DRIVE,$REG0+RPDS,$REG0+RPER,$REG0+RPCS
 6377  032046  001166
 6378  032050  001264  001124  001172  DT16:   .WORD   DRIVE,$GDDAT,$REG0+RPBA,$REG0+RPWC,$REG0+RPDS,$REG0+RPER
 6379  032056  001170  001162  001164
 6380  032064  001166                          .WORD   $REG0+RPCS
 6381  032066  001264  001162  001164  DT17:   .WORD   DRIVE,$REG0+RPDS,$REG0+RPER,$REG0+RPCS,$REG0+RPWC,$REG0+RPBA
 6382  032074  001166  001170  001172
 6383  032102  001174  001176  001200  DT17A:  .WORD   $REG0+RPCA,$REG0+RPDA,$REG0+RPM1,$REG0+20,$REG0+22
 6384  032110  001202  001204
 6385  032114  001264  001124  001126  DT23:   .WORD   DRIVE,$GDDAT,$BDDAT,$REG0+RPDS,$REG0+RPER,$REG0+RPCS
 6386  032122  001162  001164  001166
 6387  032130  001174  001202          DT23A:  .WORD   $REG0+RPCA,$REG0+20
 6388  032134  001264  001162  001164  DT24:   .WORD   DRIVE,$REG0+RPDS,$REG0+RPER,$REG0+RPCS,$REG0+RPBA
 6389  032142  001166  001172
 6390  032146  001264  001162  001164  DT25:   .WORD   DRIVE,$REG0+RPDS,$REG0+RPER,$REG0+RPCS
```

```
6391  032154  001166
6392  032156  001264  001162  001164  DT26:   .WORD   DRIVE,$REG0+RPDS,$REG0+RPER,$REG0+RPCS,$REG0+RPCA,$REG0+RPDA
6393  032164  001166  001174  001176
6394  032172  001264  001266  001270  DT27:   .WORD   DRIVE,CYL.ER,TRK.ER,SEC.ER,$REG0+RPDS,$REG0+RPER,$REG0+RPCS
6395  032200  001272  001162  001164
6396  032206  001166
6397  032210  001170  001172  001174  DT27A:  .WORD   $REG0+RPWC,$REG0+RPBA,$REG0+RPCA,$REG0+RPDA,$REG0+RPM1,$REG0+20,$REG0+22
6398  032216  001176  001200  001202
6399  032224  001204
6400  032226  001264  001124  001162  DT34:   .WORD   DRIVE,$GDDAT,$REG0+RPDS,$REG0+RPER,$REG0+RPCS,$REG0+RPDA
6401  032234  001164  001166  001176
6402  032242  001174  001202  033626  DT34A:  .WORD   $REG0+RPCA,$REG0+20,CYLDER,CYLDER+2,CYLDER+4
6403  032250  033630  033632
6404
6405  032254  000001                  DF1:    .WORD   1               ;1 'DH' LINE
6406  032256     001                          .BYTE   1               ;# OF ENTRIES IN 'DH' LINE
6407  032257     000                          .BYTE   0               ;ENTRIES ARE ALL OCTAL
6408
6409
6410  032260  000001                  DF2:    .WORD   1
6411  032262     002                          .BYTE   2               ;2 ENTRIES IN THE 'DH'
6412  032263     000                          .BYTE   0
6413
6414  032264  000002                  DF10:   .WORD   2
6415  032266     005                          .BYTE   5
6416  032267     000                          .BYTE   0
6417  032270  030466                          .WORD   DH10A
6418  032272     000                          .BYTE   0
6419  032273     000                          .BYTE   0
6420
6421  032274  000002                  DF12:   .WORD   2
6422  032276     005                          .BYTE   5
6423  032277     000                          .BYTE   0
6424  032300  030507                          .WORD   DH12
6425  032302     000                          .BYTE   0
6426  032303     000                          .BYTE   0
6427
6428  032304  000001                  DF13:   .WORD   1
6429  032306     003                          .BYTE   3
6430  032307     000                          .BYTE   0
6431
6432  032310  000001                  DF14:   .WORD   1
6433  032312     001                          .BYTE   1
6434  032313     001                          .BYTE   1               ;ITEM IS DECIMAL
6435
6436  032314  000001                  DF15:   .WORD   1
6437  032316     004                          .BYTE   4
6438  032317     000                          .BYTE   0
6439
6440  032320  000001                  DF16:   .WORD   1
6441  032322     007                          .BYTE   7
6442  032323     000                          .BYTE   0
6443
6444  032324  000002                  DF17:   .WORD   2               ;2 'DH' LINES
6445  032326     006                          .BYTE   6
6446  032327     000                          .BYTE   0
```

```
                                        .WORD   DH17A
                                        .BYTE   5
                                        .BYTE   0

                        DF23:           .WORD   2
                                        .BYTE   6
                                        .BYTE   0
                                        .WORD   DH23A
                                        .BYTE   2
                                        .BYTE   0

                        DF24:           .WORD   1
                                        .BYTE   5
                                        .BYTE   0

                        DF26:           .WORD   1
                                        .BYTE   6
                                        .BYTE   0

                        DF27:           .WORD   2               ;2 'DH' LINES
                                        .BYTE   7
                                        .BYTE   0
                                        .WORD   DH27A
                                        .BYTE   7
                                        .BYTE   0

                        DF34:           .WORD   2
                                        .BYTE   6
                                        .BYTE   0
                                        .WORD   DH34A
                                        .BYTE   5
                                        .BYTE   0


032374 047503 052116 047105 LIN11H:     .ASCII  @CONTENTS OF ERROR SECTOR (REPORTED ABOVE)@<15><12>
032402 051524 047440 020106
032410 051105 047522 020122
032416 042523 052103 051117
032424 024040 042522 047520
032432 052122 042105 040440
032440 047502 042526 006451
032446    012
032447    101 042104 020122             .ASCIZ  @ADDR    DATA@<15><12>
032454 020040 042040 052101
032462 006501 000012
032466    040                 SPACE4:   .ASCII  / /
032467    040                 SPACE3:   .ASCII  / /
032470    040                 SPACE2:   .ASCII  / /
032471    040     000         SPACE1:   .ASCIZ  / /
032473    122 030120 000062     RP02:   .ASCIZ  /RP02/
032500 050122 031460    000     RP03:   .ASCIZ  /RP03/
032505    104 044522 042526   UNTMSG:   .ASCIZ  /DRIVE /
032512 000040
032514 047440 043106 044514   UNTOFF:   .ASCIZ  / OFFLINE/
032522 042516    000
032525    040 047117 044514   UNTON:    .ASCIZ  / ONLINE/
```

```
 6503  032532  042516     000
 6504  032535     040  047516  020124  UNTNOT: .ASCIZ  / NOT BEING TESTED/
 6505  032542  042502  047111  020107
 6506  032550  042524  052123  042105
 6507  032556     000
 6508  032557     040  046101  042522  UNTASN: .ASCIZ  / ALREADY BEING TESTED/
 6509  032564  042101  020131  042502
 6510  032572  047111  020107  042524
 6511  032600  052123  042105     000
 6512  032605     040  047125  040523  NOTSAF: .ASCIZ  / UNSAFE/
 6513  032612  042506     000
 6514  032615     104  044522  042526  SYSTAT: .ASCIZ  /DRIVE STATUS:/<15><12><12>
 6515  032622  051440  040524  052524
 6516  032630  035123  005015  000012
 6517  032636  051104  053111  020105  STATHD: .ASCII  /DRIVE PERFORMANCE SUMMARY/<15><12>
 6518  032644  042520  043122  051117
 6519  032652  040515  041516  020105
 6520  032660  052523  046515  051101
 6521  032666  006531     012
 6522  032671     104  053122  050040          .ASCII  /DRV PASS ORDERS   SEEKS   WRDS READ/
 6523  032676  051501  020123  051117
 6524  032704  042504  051522  020040
 6525  032712  051440  042505  051513
 6526  032720  020040  053440  042122
 6527  032726  020123  042522  042101
 6528  032734  051440  043117  020124          .ASCIZ  / SOFT HARD SUSI MISP OTHER/<15><12>
 6529  032742  040510  042122  051440
 6530  032750  051525  020111  044515
 6531  032756  050123  047440  044124
 6532  032764  051105  005015     000
 6533  032771     007  043077  052101  DROPNG: .ASCIZ  <07>/?FATAL OR EXCESSIVE ERRORS/
 6534  032776  046101  047440  020122
 6535  033004  054105  042503  051523
 6536  033012  053111  020105  051105
 6537  033020  047522  051522     000
 6538  033025     105  042116  047440  ENDPAS: .ASCIZ  /END OF PASS/
 6539  033032  020106  040520  051523
 6540  033040     000
 6541  033041     015  042412  042116  ENDTST: .ASCIZ  <15><12>/END OF TEST/
 6542  033046  047440  020106  042524
 6543  033054  052123     000
 6544  033057     040  042504  051501  DEASSG: .ASCIZ  / DEASSIGNED/
 6545  033064  044523  047107  042105
 6546  033072     000
 6547  033073     015  025012  025052  DRNUM:  .ASCIZ  <15><12>/********** DRIVE # /
 6548  033100  025052  025052  025052
 6549  033106  020052  051104  053111
 6550  033114  020105  020043     000
 6551  033121     040  052123  051101  ASGND:  .ASCIZ  / STARTED/<15><12>
 6552  033126  042524  006504  000012
 6553  033134  000077                  QUES:   .ASCIZ  /?/
 6554  033136  047111  040526  044514  INVLD:  .ASCIZ  /INVALID COMMAND/<15><12>
 6555  033144  020104  047503  046515
 6556  033152  047101  006504  000012
 6557  033160  020077  047111  040526  BADENT: .ASCIZ  /? INVALID ENTRY/<15><12>
 6558  033166  044514  020104  047105
```

```
 6559  033174  051124  006531  000012
 6560  033202  005015  051120  043517  INTDON: .ASCIZ  <15><12>/PROGRAM INITIALIZATION COMPLETE, ENTER COMMANDS/<15><12>
 6561  033210  040522  020115  047111
 6562  033216  052111  040511  044514
 6563  033224  040532  044524  047117
 6564  033232  041440  046517  046120
 6565  033240  052105  026105  042440
 6566  033246  052116  051105  041440
 6567  033254  046517  040515  042116
 6568  033262  006523  000012
 6569                                  .EVEN
 6570
 6571  033266  033561                  PARLST: .WORD   PAR11            ;PARAMETER ENTRY LIST
 6572  033270  001216                          .WORD   RPADR
 6573  033272  033570                          .WORD   PAR12
 6574  033274  001220                          .WORD   RPVEC
 6575  033276  033471                          .WORD   PAR1
 6576  033300  001306                          .WORD   BUFRSZ          ;LOCATION
 6577  033302  033521                          .WORD   PAR4
 6578  033304  001324                          .WORD   LC
 6579  033306  033525                          .WORD   PAR5
 6580  033310  001326                          .WORD   FC
 6581  033312  033531                          .WORD   PAR6
 6582  033314  001330                          .WORD   LT
 6583  033316  033535                          .WORD   PAR7
 6584  033320  001332                          .WORD   FT
 6585  033322  033541                          .WORD   PAR8
 6586  033324  001334                          .WORD   LS
 6587  033326  033545                          .WORD   PAR9
 6588  033330  001336                          .WORD   FS
 6589  033332  033501                          .WORD   PAR2            ;PARAMETER NAME
 6590  033334  001312                          .WORD   INTRVL
 6591  033336  033616                          .WORD   PAR19
 6592  033340  001304                          .WORD   PASCNT
 6593  033342  033551                          .WORD   PAR10
 6594  033344  001316                          .WORD   SEKMOD
 6595  033346  033577                          .WORD   PAR14
 6596  033350  001320                          .WORD   RATIO
 6597  033352  033511                          .WORD   PAR3
 6598  033354  001314                          .WORD   CMPLMT
 6599  033356  033606                          .WORD   PAR15
 6600  033360  001322                          .WORD   AUTOCK
 6601  033362  000000                          .WORD   0               ;TABLE TERMINATOR
 6602
 6603  033364  047105  042524  020122  ASKPAR: .ASCIZ  /ENTER PARAMETERS:/<15><12><12>
 6604  033372  040520  040522  042515
 6605  033400  042524  051522  006472
 6606  033406  005012     000
 6607  033411     015  042412  051122  NGPAR:  .ASCIZ  <15><12>/ERROR IN PARAMETER:  /
 6608  033416  051117  044440  020116
 6609  033424  040520  040522  042515
 6610  033432  042524  035122  020040
 6611  033440     000
 6612  033441     040  051040  026505  REPAR:  .ASCIZ  /  RE-ENTER PARAMETERS/<15><12>
 6613  033446  047105  042524  020122
 6614  033454  040520  040522  042515
```

```
 6615  033462  042524  051522  005015
 6616  033470     000
 6617
 6618  033471     102  043125  051522  PAR1:   .ASCIZ  /BUFRSZ /
 6619  033476  020132     000
 6620  033501     111  052116  053122  PAR2:   .ASCIZ  /INTRVL /
 6621  033506  020114     000
 6622  033511     103  050115  046514  PAR3:   .ASCIZ  /CMPLMT /
 6623  033516  020124     000
 6624  033521     114  020103     000  PAR4:   .ASCIZ  /LC /
 6625  033525     106  020103     000  PAR5:   .ASCIZ  /FC /
 6626  033531     114  020124     000  PAR6:   .ASCIZ  /LT /
 6627  033535     106  020124     000  PAR7:   .ASCIZ  /FT /
 6628  033541     114  020123     000  PAR8:   .ASCIZ  /LS /
 6629  033545     106  020123     000  PAR9:   .ASCIZ  /FS /
 6630  033551     123  045505  047515  PAR10:  .ASCIZ  /SEKMOD /
 6631  033556  020104     000
 6632  033561     122  040520  051104  PAR11:  .ASCIZ  /RPADR /
 6633  033566  000040
 6634  033570  050122  042526  020103  PAR12:  .ASCIZ  /RPVEC /
 6635  033576     000
 6636  033577     122  052101  047511  PAR14:  .ASCIZ  /RATIO /
 6637  033604  000040
 6638  033606  052501  047524  045503  PAR15:  .ASCIZ  /AUTOCK /
 6639  033614  000040
 6640  033616  040520  041523  052116  PAR19:  .ASCIZ  /PASCNT /
 6641  033624  000040
 6642
 6643                                           .EVEN
 6644
 6645  033626  000000  000000  000000  CYLDER: .WORD   0,0,0           ;HEADER BUFFER FOR 'READHD' ROUTINE
 6646
 6647          033634                  ENDPGM  =       .               ;BUFFERS START HERE
 6648
 6649  033634  005015  041412  051132  TITLE:  .ASCII  <15><12><12>/CZRP1C/<15><12>
 6650  033642  030520  006503     012
 6651  033647     122  030520  042461          .ASCIZ  /RP11E MULTI-DRIVE EXERCISER PROGRAM/<15><12><12>
 6652  033654  046440  046125  044524
 6653  033662  042055  044522  042526
 6654  033670  042440  042530  041522
 6655  033676  051511  051105  050040
 6656  033704  047522  051107  046501
 6657  033712  005015  000012
 6658                                           .EVEN
 6659
 6660                                   .SBTTL  ROUTINE TO SIZE MEMORY
 6661
 6662                                   ;;*****************************************************************
 6663                                   ;*CALL:
 6664                                   ;*      JSR     PC,$SIZE
 6665                                   ;*      RETURN
 6666                                   ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
 6667
 6668  033716  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
 6669  033720  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
 6670  033722  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
```

ROUTINE TO SIZE MEMORY SEQ 0123

```
 6671  033726  013746  000006                MOV     @#ERRVEC+2,-(SP)
 6672  033732  010600                        MOV     SP,R0                   ;;SAVE THE STACK POINTER
 6673                                ;;SET THE ERRVEC PS TO THE PRESENT PS
 6674  033734  104400                        TRAP                            ;;PUSH OLD PSW AND PC ON STACK
 6675  033736  012637  000006                MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
 6676  033742  012737  033762  000004        MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
 6677  033750  012701  020000                MOV     #20000,R1               ;;FIRST ADDRESS
 6678  033754  005711                1$:     TST     (R1)                    ;;TEST THIS ADDRESS
 6679  033756  005721                        TST     (R1)+                   ;;STEP TO NEXT ADDRESS
 6680  033760  000775                        BR      1$                      ;;TRY ANOTHER
 6681  033762  162701  000002        2$:     SUB     #2,R1                   ;;DROP BACK
 6682  033766  010006                        MOV     R0,SP                   ;;RESTORE THE STACK
 6683  033770  012637  000006                MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
 6684  033774  012637  000004                MOV     (SP)+,@#ERRVEC
 6685  034000  010137  034012                MOV     R1,$LSTAD               ;;LAST ADDRESS
 6686  034004  012601                        MOV     (SP)+,R1                ;;RESTORE R1
 6687  034006  012600                        MOV     (SP)+,R0                ;;RESTORE R0
 6688  034010  000207                        RTS     PC
 6689  034012  000000        $LSTAD: .WORD   0                       ;;CONTAINS THE LAST ADDRESS
 6690
 6691          000001                        .END
```

```
ABNRML  011276          2133    2989#
ACTDRV  021120          4718#   4862*   4871*   5002*   5012*   5 35
ACTSTR  021121          4724#   5137*   5157*
AIE   = 020000          1237#
ASGND   033121          2041    6551#
ASGN0   010420          2827#   2926    2932
ASGN1   010434          2831#
ASGN2   010472          1949    2830    2839#   2842
ASGN3   010514          2836    2839    2844#   2865
ASGN4   010616          2843    2845    2862#
ASGN5   010626          2863    2866#
ASGN6   010634          2851    2868#
ASGN7   010644          2852    2870#
ASGN8   010652          2867    2869    2871#
ASKPAR  033364          1790    6603#
ASNERR  011110          2837    2893    2911    2936#
ASNLST  025450          1744    1991    2048*   2575    2583    2834    2844    2880    2882*   2904    2914    2973*   3048*
                        5696#
ASNMSG  011130          2832*   2868*   2870*   2892*   2910*   2942#
ATABIT  021274          1916    2048    2575    2583    2834    2844    2880    2882    2904    2973    3048    4769#   4797
                        4844    4846    4924    4946    494?    4973    5019    5027    5098    5099    5106    5113    5126
                        5179    5215    5216
AUTOCK  001322          1375#   2325    2428    5051    6600
AVAIL   025540          1994    2042    2046*   2049    2069    2071    2096    2097    2136    5710#
BADENT  033160          6557#
BADOP   013576          3463    3481#
BADPAR  003256          1849    1856    1860    1864    1868    1872    1876    1880    1884#
BDADR   014076          3522    3555#
BEGCOD  001342          1401#   2053
BEGIN   002032          1694    1696#
BEGPAT  001340          1400#   2054
BEGSIZ  001344          1403#   1850    1852*   2056    2057
BIT0  = 000001          1153#   5256
BIT00 = 000001          1143#   1153
BIT01 - 000002          1142#   1152
BIT02   000004          1141#   1151    2149    2265    2435    3298    3575    3737    5053
BIT03 - 000010          1140#   1150
BIT04 - 000020          1139#   1149
BIT05 - 000040          1138#   1148
BIT06 - 000100          1137#   1147    3458    3500    5046
BIT07 - 000200          1136#   1146    3450    4906    5025    5123
BIT08   000400          1135#   1145    3456    5183
BIT09 - 001000          1134#   1144    3466    5205
BIT1  - 000002          1152#
BIT10 - 002000          1133#   3893
BIT11   004000          1132#
BIT12 - 010000          1131#   3464    4994    5044    5115    5189    5207
BIT13   020000          1130#   3462    3900    5276
BIT14 - 040000          1129#   3460    4992    5042    5118    5187    5210
BIT15 - 100000          1128#   4992    4994    5042    5044    5046    5115    5118    5121    5183    5205    5207    5210
                        5276
BIT2    000004          1151#
BIT3  - 000010          1150#
BIT4  - 000020          1149#
BIT5    000040          1148#   5121
BIT6    000100          1147#
```

```
BIT7  = 000200           1146#   3816    5121
BIT8  = 000400           1145#   3468
BIT9  = 001000           1144#
BLKADR  025726           1746    2854    2855    2884    2907    5740#
BLK0    024264           2574    5364#   5741
BLK1    024376           5404#   5742
BLK2    024510           5444#   5743
BLK3    024622           5484#   5744
BLK4    024734           5524#   5745
BLK5    025046           5564#   5746
BLK6    025160           5604#   5747
BLK7    025272           5644#   5748
BPTVEC= 000014           1160#
BUFRSZ  001306           1358#   1778    1780*   1781    1783*   1844    1846    1850    1852    2273    2378    2486    2487
                         6576
BUFTBL  025604           1765*   1766*   1767*   1768*   1770*   1775*   1777*   1781    1783    1844    2172    2174    2192*
                         2208    2209    2221*   2227*   2228    2238    2239    2252*   5718#
CHNGPM  001262           1341#   1693*   1695*   1784    1843*
C1      021636           4868    4877#   5010    5155
CIR     022426           4919    4941    4962    4988#
CI1     021750           4883    4903#   4910
CI2     022014           4886    4915#   4932
CI3     022132           4889    4937#   4954
CI4     022254           4892    4959#   4983
CKADR   002710           1785    1824#
CKBUS   012456           3260    3281#
CKCLK   006752           1930    2545#   3459    3501
CKCLK1  007034           2546    2556#
CKCLK2  007102           2557    2565#
CKCLK3  007106           2555    2564    2566#
CKERR   012424           3259    3268#
CKHNF   014450           3534    3637#
CKPAR   003076           1851    1853#
CKSWR = 104407           3868    3908    4654#
CK.CHR  011764           3121#   3147    3164
CK.NUM  012010           1804    3143#
CK.OCT  011734           3102#   3145    3159
CLEAR - 000001           1186#   5037    5190    5191
CLKFLG  001236           1330#   2545*   2549*   2559*   2710
CLOCK   007650           2551    2561    2719#
CLRDPB  011140           2856    2948#
CMBUF   013154           3309*   3380#   6372
CMCNT   013144           3302*   3303*   3315    3317    3320*   3355    3376#
CMCYL   013146           3304*   3365*   3377#   6372
CMDAT   012702           3321#
CMPAR   012524           3263    3296#   3612    3702
CMPARD  012546           3299    3301#
CMPLMT  001314           1364#   3307    6598
CMPRES  006640           2019    2099    2118    2140    2320#   2523
CMPRT   013156           3333    3342    3352    3384#
CMPRT1  013264           3395    3401    3404#
CMPRX   013130           3356    3367#
CMSEC   013150           3306*   3357*   3358    3360*   3378#   6372
CMSTR   012610           3307#   3359    3363    3366
CMTRK   013152           3305*   3361*   3362    3364*   3379#   6372
CNTRLC  001242           1332#   1789*   1933*   4171    4254
```

```
CNTRLR  014106            3525    3561#
COMTBL  025746            2344    2439    5750#
CR    = 000015            1068#   3972    3982
CRLF  = 000200            1069#   3943    3982
CSME  = 000040            1216#   3535    3689
CYLDER  033626            3643    5684    6402    6645#
CYL.ER  001266            1343#   3876*   6394
DATAPK  011064            2814    2925#
DATA0   026020            5781    5784#
DATA1   026060            5766    5767    5768    5769    5770    5771    5772    5773    5774    5775    5776    5777    5778
                          5779    5780    5782    5801#
DATER   014612            3537    3665#
DDISP = 177570            1075#   1296    1716
DEASGN  010654            2804    2875#
DEASSG  033057            2015    6544#
DF1     032254            1429    6405#
DF10    032264            1478    6414#
DF12    032274            1492    6421#
DF13    032304            1499    6428#
DF14    032310            1506    6432#
DF15    032314            1513    1569    1575    6436#
DF16    032320            1520    6440#
DF17    032324            1527    1534    1541    1548    1640    1668    6444#
DF2     032260            1436    1443    1485    6410#
DF23    032334            1555    6451#
DF24    032344            1562    6458#
DF26    032350            1576    6462#
DF27    032354            1583    1604    1611    1625    1682    6466#
DF34    032364            1618    1633    6473#
DH1     030342            1427    6220#
DH10    030402            1476    1490    6227#
DH10A   030466            6236#   6417
DH12    030507            6239#   6424
DH14    030561            1504    6247#
DH15    030607            1511    1673    6251#
DH16    030644            1518    6256#
DH17    030765            1525    1532    1539    1546    1638    1666    6270#
DH17A   031042            6278#   6447
DH2     030350            1434    6221#
DH23    031107            1553    6285#
DH23A   031212            6297#   6454
DH24    031227            1560    6300#
DH25    031274            1567    6307#
DH26    031331            1574    6312#
DH27    031406            1581    1602    1609    1623    1680    6320#
DH27A   031473            6329#   6469
DH3     030365            1441    6224#
DH30    031560            1588    6338#
DH31    031573            1595    6340#
DH34    031606            1616    1631    6342#
DH34A   031663            6350#   6476
DH41    031735            1652    6358#
DH42    031755            1659    6361#
DISPLA  001142            1296#   1716*   1724*   3892*
DISPRE  000174            1248#   1724
DONE    013652            3452    3506#
```

```
DORTI   006630          1933    2515#
DRIVE   001264          1342#   3871*   6369    6372    6376    6378    6381    6385    6388    6390    6392    6394    6400
DRNUM   033073          2979    3020    3026    6547#
DROP    011216          2970#   2993    3477    3488    3494    3502
DROPNG  032771          2978    6533#
DRVACT  021100          4677#   4895    4924*   4946*   4973*   5019*   5074    5082    5098*   5179*   5215*
DRVINT  021434          4806    4828#   4988    5038    5182
DRVQUE  024216          5057    5110    5279    5294#
DRVSTA  021102          1906    2850    4684#   4829*   4839*   4841*   4989    5039    5116*   5119*   5184    5208*   5211*
DRVTYP  021112          1836    1916    4698#   4844*   4846*
DSKERR= 000001          1211#   3599    3616
DSWR  = 177570          1074#   1295    1715
DT1     032000          1428    6368#
DT10    032016          1477    1491    6372#
DT11    032012          1484    6371#
DT13    032030          1498    6374#
DT14    032036          1505    6375#
DT15    032040          1512    1674    6376#
DT16    032050          1519    6378#
DT17    032066          1526    1533    1540    1547    1639    1667    6381#
DT17A   032102          6383#
DT2     032002          1435    6369#
DT23    032114          1554    6385#
DT23A   032130          6387#
DT24    032134          1561    6388#
DT25    032146          1568    6390#
DT26    032156          1575    6392#
DT27    032172          1582    1603    1610    1624    1681    6394#
DT27A   032210          6397#
DT3     032006          1442    6370#
DT34    032226          1617    1632    6400#
DT34A   032242          6402#
DUMIT   025474          1985    2884*   2975*   3050*   5702#
EMTVEC= 000030          1163#   1705*   1706*   1727*
EM1     027020          1426    6062#
EM10    027151          1475    6081#
EM12    027216          1489    6089#
EM15    027242          1510    6094#
EM16    027315          1517    6103#
EM17    027371          1524    6112#
EM2     027062          1433    6069#
EM20    027435          1531    6120#
EM21    027471          1538    6126#
EM22    027507          1545    6130#
EM23    027524          1552    6134#
EM24    027544          1559    6138#
EM25    027572          1566    6143#
EM26    027613          1573    6147#
EM27    027645          1580    6153#
EM3     027107          1440    6074#
EM32    027660          1601    6156#
EM33    027722          1608    6163#
EM34    027772          1615    6171#
EM35    030013          1622    6175#
EM36    030073          1629    6185#
EM37    030161          1637    6196#
```

```
EM43    030216            1665    6202#
EM44    030257            1672    6209#
EM45    030312            1679    6215#
ENDCMP  013352            3354    3420#
ENDCON  001274            1353#   3000    3003
ENDPAS  033025            3012    6538#
ENDPGM= 033634            1766    1768    2269    6647#
ENDSEK  001300            1355#   3006    3009
ENDTST  033041            3019    6541#
ENTPAR  002556            1789#   1831    1889
ENTPR   002564            1790#   1817
ENTPR1  002662            1789    1815#
ENTPR2  002674            1793    1809    1814    1818#
EOP   = 000002            1212#   2465    3506
ERCTR   013140            3312*   3329    3374#   3396*   3402*   3420    3719*   6375
ERPRC1  013466            3451    3456#
ERPROC  013452            3258    3450#
ERR   = 100000            1239#   3641    5104
ERRVEC= 000004            1156#   1713    1714*   1725*   1824*   1826*   1828*   2546*   2547*   2557*   2566*   6670    6671
                          6675*   6676*   6683*   6684*
FACTOR  004706            2145*   2146*   2148*   2151    2159#
FAIRNS  001254            1337#   2059    2420    2496
FC      001326            1388#   1857    2052    2369    2373    2480    6580
FILBUF  005274            2081    2110    2264#   3688
FMTE  - 001000            1220#   3529    3599    3616
FMTERR  014126            3531    3573#
FRSTER  013133            3311*   3344    3370#   3384    3388*   3722*
FS      001336            1392#   1873    2050    2349    2353    2479    6588
FT      001332            1390#   1865    2051    2359    2363    2478    6584
FUV   - 040000            1225#   3514
GENDPB  025404            2848*   3777*   3778*   3780    3782    3789*   3790*   3791*   3792*   3793*   3794*   3796    3798
                          5684#
GENREG  025424            3641    5686    5688#
GETBUF  004710            2078    2106    2169#   3597    3684
GETCYL  005740            2368#
GETPAR  005502            2076    2318#
GETPAT  006274            2418    2444#   2449    2494
GETREM  011570            2334    2352    2362    2372    2380    2427    2445    3055#
GETRK   005710            2358#
GETSEC  005660            2339    2348#
GETSIZ  005770            2378#   2391
GNS   - ****** U          1247    4646    4647    4648    4649    4650    4652    4654    4655    4656    4657    4658
GODRIV  005424            2083    2111    2296#   3810
GTPAR1  005556            2322    2329#
GTSWR   104406            1738    4652#
HDR   = 004000            1235#   3793
HE    - 040000            1238#   5104
HNF   = 010000            1204#   3268    3532
HOMSEK  000015            1192#   3778    5262
HOUR    010120            1750*   2712    2741*   2742    2744    2746*   2747*   2749*   2761#
HT    = 000011            1066#   3941    3982
HZ      001240            1331#   1748    2722
IDE     000100            1232#   4971    5033
IDLE    004522            2070    2109    2123#
INCHRD  007534            2675#   3608    3697
INCMIS  007570            2693#   3654
```

```
INCSKI  007552            2684#   3631
INCSOF  007516            2666#   3611    3701
INCTOT  007606            2702#   3273    3290    3423    3476    3487    3493    3499    3544    3550    3556    3562    3568
                          3601    3615    3630    3660    3671
INTDON  033202            1929    6560#
INTRFA  014116            3528    3567#
INTRVL  001312            1362#   1753*   2732*   2752    2754    2756*   6590
INVLD   033136            2820    6554#
IOTVEC= 000020            1161#
ISR     022472            4802    5002#
KIPAR0= ****** U          6666
KSR     010136            1934    2773#   2787
KSR1    010220            2775    2786#
KSR2    010410            2782    2801    2805    2809    2815    2819    2821#
LC      001324            1384#   1840    1842*   1853    1857    2368    2473    6578
LF    = 000012            1067#   3976    3982
LIMIT   013142            3307*   3308*   3375#   3404    3406*
LIN11H  032374            3752    6481#
LODQUE  024050            4866    5255#
LPE   = 000200            1218#   3535    3689
LS      001334            1391#   1869    1873    2348    6586
LSTAD   001244            1333#   1764*   1767    1771
LT      001330            1389#   1861    1865    2358    2476    6582
MAIN    003650            1975#   2141    2485
MAIN0   003676            1976    1978    1984#
MAIN1   004046            1951    1957    1992    2024#
MAIN2   004266            2034    2066#   2119
MAIN3   004436            2067    2104#
MASK    015562            3599*   3616*   3689*   3818    3830#
MATCH   013374            3321    3430#
MAXCYL  001256            1338#   1835*   1839*   1842    1853
MAXER   001310            1361#   2991
MINUTE  010124            1751*   2713    2733*   2734    2736*   2737*   2738    2740*   2763#
MODE  = 010000            1236#   3523    3793
MODERR- 000400            1219#
MONTR   003532            1939#
NEWASN  010414            2800    2826#
NEWUNT  025516            2028    2046    2047*   2854*   5706#
NGPAR   033411            1885    6607#
NOMTCH  015030            3323    3708#
NOTSAF  032605            1914    2870    6512#
NRML    011324            2134    2998#
NRMLX   011566            3005    3008    3010    3044    3051#
NRML1   011356            2999    3002    3006#
NRML2   011400            2481    3001    3004    3007    3011#
NXC   = 020000            1224#   3520
NXME  - 000004            1213#   3526
NXS   = 004000            1222#   3520
NXT   = 010000            1223#   3520
OFLIN   013564            3461    3475#
ORDERQ  025452            2084    2113    2123    5698#
PACK    001252            1336#   1755*   2826*   2858    2860    2925*   2931*
PARLST  033266            1791    6571#
PARXIT  006206            2328    2340    2420#
PAR1    033471            1848    6575    6618#
PAR10   033551            6593    6630#
```

```
PAR11   033561          6571    6632#
PAR12   033570          6573    6634#
PAR14   033577          1879    6595    6636#
PAR15   033606          6599    6638#
PAR19   033616          6591    6640#
PAR2    033501          6589    6620#
PAR3    033511          6597    6622#
PAR4    033521          1855    6577    6624#
PAR5    033525          1859    6579    6625#
PAR6    033531          1863    6581    6626#
PAR7    033535          1867    6583    6627#
PAR8    033541          1871    6585    6628#
PAR9    033545          1875    6587    6629#
PASCNT  001304          1357#   3017    6592
PATCOD  006176          2418#
PIRQ  = 177772          1073#
PIRQVE= 000240          1167#
POPQUE  024246          4908    4930    4952    4979    5311#   5314
PREVST  006606          2319    2468    2501#
PROCES  012364          2131    3256#
PROG  = 002000          1221#   3523
PRTBAD  015134          3670    3698    3731#
PR0   = 000000          1090#
PR1   = 000040          1091#
PR2   = 000100          1092#
PR3   = 000140          1093#
PR4   = 000200          1094#   1931    2774
PR5   = 000240          1095#
PR6   = 000300          1096#
PR7   - 000340          1097#
PS    = 177776          1070#   1071    1731*   4790    4791*   4811*   4814*   4872*
PSW   = 177776          1071#   1754*   1816*   1819*   1931*   1935*   4860    4861*
PWRVEC= 000024          1162#
QDONE   021212          4745#   4890    4959    4960    4961    4978    4982    5109
QREGS   021124          4736#   4881    4903    4904    4907    4909    5258
QSEEK   021146          4739#   4884    4915    4916    4918    4927    4929    4931    5264
QTRANS  021170          4742#   4887    4937    4939    4940    4950    4951    4953    5056    5278
QUES    033134          2936    6553#
QWAIT   021234          4748#   4927*   4950*   5089    5095*   5195    5213*
RANWRT  006220          2337    2426#
RATIO   001320          1366#   1877    2335    6596
RDCHR   104410          4330    4655#
RDLIN   104411          1802    2786    4656#
RDSEK - 000005          1188#   2411    2491    5266    5753
RDY   = 000200          1233#
READ  = 000017          1193#   2398    3794    5272    5754
READDR  006656          2502    2528#   3586    3673    3873
READHD  015372          3640    3789#
RECALT  015340          3604    3632    3655    3693    3777#
RECALO  015344          2849    3778#
REDAPK  011076          2818    2931#
RELBUF  005046          2132    2207#   3593    3680
REPAR   033441          1888    6612#
RESREG= 104413          2233    2291    2594    2606    2821    3245    4591    4658#   4815    4847    4898    5011    5156
                        5244
RESVEC= 000010          1157#
```

```
RETRY   015462          3602    3605    3618    3691    3694    3809#
RPADR   001216          1322#   1825    4804    4878    5004    5168    5228    6368    6572
RPBA  = 000010          1175#   4965*   6378    6381    6388    6397
RPCA  = 000012          1176#   4921*   4943*   4966*   6383    6387    6392    6397    6402
RPCS  = 000004          1173#   4830*   4897*   4903*   4915*   4923*   4937*   4945*   4959*   4972*   5026*   5028    5033*
                        5037*   5072    5076*   5091*   5104    5169*   5177    5190*   5191*   5229*   6369    6376    6380
                        6381    6385    6388    6390    6392    6394    6400
RPDA  = 000014          1177#   4922*   4944*   4967*   6383    6392    6397    6400
RPDS  = 000000          1171#   4831    4833    4835    4837    4842    5027*   5069    5071    5092    5099*   5100    5102
                        5197    5199    5201    5203    6376    6378    6381    6385    6388    6390    6392    6394    6400
RPER  = 000002          1172#   3641    6376    6378    6381    6385    6388    6390    6392    6394    6400
RPERRS  021074          4670#   4792    5071*   5072*   6370
RPINIT  021304          1833    4789#
RPM1  = 000016          1178#   6383    6397
RPM2  = 000020          1179#   5233
RPM3  = 000022          1180#
RPPRIO  001222          1324#   4791    4803    4861
RPTMR   023350          2751    5135#
RPVEC   001220          1323#   4801    6574
RPWC  = 000006          1174#   4964*   6378    6381    6397
RPZERO  011646          3066    3068    3070    3072#
RPZRC   011732          3065*   3067*   3069*   3071*   3084    3091#
RP02    032473          1915    6496#
RP03    032500          1918    6497#
RP11    021556          2298    3779    3795    4860#
SAVEFG  021122          1834*   4732#   5030    5124
SAVER1  013134          3313*   3347*   3372#   3389    3393    3397*
SAVER5  013136          3314*   3348*   3373#   3394    3398*
SAVREG= 104412          2207    2264    2571    2599    2773    3182    4565    4657#   4789    4828    4877    5003    5138
                        5227
SC      023024          5009    5069#
SCMND   010762          2808    2900#
SC0     023066          5035    5043    5045    5047    5070    5080#   5154
SC1     023074          5082#   5086
SC2     023102          5084#   5093    5114    5125    5128
SC3     023112          5083    5088#
SDETAL  007276          2589    2605    2618#
SECOND  010130          1752*   2714    2724*   2725    2727*   2728*   2729    2731*   2765#
SEC.ER  001272          1345#   3875*   6394
SEEK  = 000011          1190#   4945    5260
SEEKWT  021116          4712#   4947*   5106    5113*   5126*   5216*
SEKMOD  001316          1365#   2998    6594
SETVEC  003302          1878    1894#
SHDTYP  007260          2582    2600    2611#
SILO  = 000026          1182#   5242
SIXTEE  010134          1748*   1749*   2720*   2722*   2723*   2769#
SIZMEM  002370          1759#   1762*   1763*
SKIER   014414          3519    3627#
SPACE1  032471          1801    1912    2639    2648    6495#
SPACE2  032470          2625    2629    2634    3223    3232    3243    3760    6494#
SPACE3  032467          2621    6493#
SPACE4  032466          1905    1921    6492#
STACK   001100          1061#   1334    1703    1815    1820
STACK1  001246          1334#
START   002016          1266    1693#   1967
START1  002024          1267    1695#
```

```
STATHD  032636          2613    6517#
STATIN  001250          1335#   1742*   1975    1979*   2757*
STATIS  004616          2130    2145#
STATPR  007116          1980    2571#   2920
STKLMT= 177774          1072#
STNDAT  025754          2281    2289    3434    3442    5765#
STO     023462          5145    5168#
STO1    023626          5174    5194#
STO2    023706          5198    5207#
STO3    023724          5202    5210#
STO4    023736          5206    5209    5212#
STO5    023766          5176    5178    5193    5200    5204    5218#
SUCA  = 000024          1181#
SUFU  = 001000          1201#   3268    4831    5100    5197
SUOL  = 040000          1206#   4833    5102    5201
SURDY = 100000          1207#   4837    5203
SURP03- 020000          1205#   4842
SUSI  = 004000          1203#   3268    3517    4835    5199
SUSU  = 002000          1202#   5092
SUWP  = 000400          1200#
SVRP11  023770          4905    4995    5032    5036    5127    5181    5217    5227#
SWR     001140          1295#   1701    1715*   1717    1723*   1736    1896    1977    2321    2331    2463    2482    2780
                        2812    2989    3015    3183    3261    3331    3408    3724    3731    3733    3753    3768    3893
                        3900    3905    4174    4213    4268*
SWREG   000176          1249#   1723    1736    2780    4174    4213    4236
SW0   - 000001          1125#   2331    2812
SW00  = 000001          1115#   1125
SW01  - 000002          1114#   1124    3261
SW02  = 000004          1113#   1123    3408
SW03  = 000010          1112#   1122    3753
SW04  = 000020          1111#   1121    2482    2989    3015
SW05  - 000040          1110#   1120    2321    2463
SW06  = 000100          1109#   1119    1977
SW07  = 000200          1108#   1118    1896
SW08    000400          1107#   1117
SW09  - 001000          1106#   1116
SW1   = 000002          1124#
SW10  - 002000          1105#
SW11  - 004000          1104#
SW12  - 010000          1103#
SW13    020000          1102#   3183    3733
SW14    040000          1101#
SW15    100000          1100#   3724    3768
SW2   = 000004          1123#
SW3   = 000010          1122#   3331    3731
SW4   = 000020          1121#
SW5   = 000040          1120#
SW6   = 000100          1119#
SW7     000200          1118#
SW8   = 000400          1117#
SW9     001000          1116#
SYSTAT  032615          1901    6514#
TBITVE- 000014          1158#
TD      022542          5007    5017#
TIMEE   000020          1215#   3526
TIMER   021254          4756#   4793    4926*   4949*   4975*   5021*   5096*   5141    5143*   5171*
```

```
TIMIO   013626          3469    3498#
TIMOUT  001260          1340#   4926    4949    4975
TIMPOS  013614          3467    3492#
TITLE   033634          1729    6649#
TKVEC = 000060          1165#   1934*   2784*   2787*   4151*   4152*
TPVEC = 000064          1166#
TRAPVE= 000034          1164#   1707*   1708*   1728*
TRK.ER  001270          1344#   3874*   6394
TRNSWT  021114          2846    4704#   4879    4893    4977*   4980    5005    5018    5023    5024*   5152    5173    5175
                        5180    5192*
TRTVEC= 000014          1159#
TYPDS = 104405          3014    3220    4650#
TYPE  = 104401          1729    1790    1795    1796    1801    1885    1886    1888    1900    1901    1905    1919    1921
                        1922    1924    1929    2009    2011    2015    2035    2037    2041    2611    2613    2621    2625
                        2629    2634    2639    2648    2659    2712    2713    2714    2790    2820    2936    2937    2941
                        2943    2976    2977    2978    2979    2983    3011    3012    3019    3020    3024    3026    3030
                        3045    3085    3194    3196    3200    3201    3210    3223    3227    3232    3234    3240    3243
                        3752    3760    3765    3767    3895    3903    3946    4043    4119    4168    4180    4234    4235
                        4238    4249    4259    4270    4289    4337    4343    4348    4352    4357    4358    4360    4363
                        4367    4646#
TYPERR  012120          3182#   3387    3415    3710    3715    3902
TYPEST  007230          2016    2599#   2908    3031
TYPOC = 104402          1800    3216    3757    3762    4237    4647#
TYPON = 104404          4649#
TYPOS = 104403          1903    2013    2039    2619    2939    2981    3022    3028    4648#
UNSAF   013602          3465    3486#   3516
UNTASN  032557          2832    6508#
UNTMSG  032505          2011    2037    2937    6498#
UNTNOT  032535          2892    2910    6504#
UNTOFF  032514          1911    2868    6500#
UNTON   032525          1909    6502#
WAIT    025562          2001    2066    2093    2104    2117    5714#
WC      022732          5034    5051#
WCE   = 000010          1214#   3508    3665    3689
WCEER   014056          3510    3543#
WPE   = 000100          1217#   3535    3689
WPV   = 100000          1226#   3511
WPVER   014066          3513    3549#
WRITE = 000013          1191#   2396    5274    5751
WRTCHK= 000007          1189#   2267    2323    2338    3296    3573    3686    3735    5063    5270    5752
WRTPK   006322          2074    2455#
WRTPK1  006352          2457    2459    2463#
WRTPK2  006550          2462    2491#
WRTPK3  006560          2461    2490    2493#
WRTPK4  006576          2464    2492    2496#
WRTSEK= 000003          1187#   2409    2493    5268    5750
ZROIND  013132          3310*   3341*   3346*   3350    3369#
$AUTOB  001134          1292#   1740*   4227    4384
$BDADR  001122          1287#   3410*   3411*   3712*   6371    6374
$BDDAT  001126          1289#   3413*   3628*   3643*   3644*   3645*   3646*   3647*   3648*   3649*   3650    3713*   6371
                        6374    6385
$BELL   001206          1317#   3895    3911    4180    4377
$BUF  - 000006          2079*   2107*   2146    2213    2219    2224    2235    2269    2277    3283    3301    3578    3582
                        3598*   3685*   5329#   5330
$CHARC  016260          3948*   3958*   3965    3974*   3979#
$CKSWR  017234          4213#   4654
```

```
$CMTAG  001100          1275#  1698   1699   1711
$CM1   = 000012         1307#  1308#  1309#  1310#  1311#  1312#  1313#  1314#  1315#  1316#  1317#
$CM2   = 000024         1307#  1308#  1309#  1310#  1311#  1312#  1313#  1314#  1315#  1316#  1317#
$CM3   = 000012         1305#  1307
$CNTLC  020134          4168   4249   4377#
$CNTLG  020146          4234   4379#
$CNTLU  020141          4259   4352   4378#
$COMND- 000002          2053*  2149   2265   2267   2323   2327*  2329   2338   2344*  2392   2396   2398   2405
                        2407   2409*  2411*  2439*  2467   2491*  2493*  3296   3298   3573   3575   3686   3735
                        3737   3778*  3793*  3794*  5053   5060   5063*  5326#  5327
$CRLF   001213          1319#  1730   1795   1900   1924   2009   2035   2611   2659   2790   2943   2976   2977
                        2983   3011   3024   3030   3045   3194   3200   3210   3227   3240   3765   3767   3903
                        3911   3947   3982   4270   4357   4377
$CYL   - 000010         2052*  2153   2304   2374*  2472*  2473   2480*  3304   3589*  3628   3657*  3676*  3792*
                        5061   5330#  5331
$DBLK   016726          4085   4119   4127#
$DB2D   020614          2627   2632   2637   3845   4565#
$DECVL  020774          4567   4613#
$DIV    020176          2386   3058   3744   4411#
$DOAGN  003644          1961   1967#
$DTBL   016716          4088   4123#
$ENDAD  003634          1257   1832   1958   1959   1963#
$ERFLG  001103          1278#  3890*  3911
$ERMAX  001115          1284#
$ERROR  0156'4          1705   3867#
$ERRPC  001116          1285#  3897*  3898*  3899   3911
$ERRTB  001346          1422#  3191
$ERTTL  001112          1282#  3896*  3911
$FAIR  = 000064         2059*  2082*  2089*  2112*  2420*  2496*  5349#  5350
$FILLC  001156          1303#  3951   3982
$FILLS  001155          1302#  3982
$GDADR  001120          1286#
$GDDAT  001124          1288#  3284*  3412*  3627*  3652*  6374   6378   6385   6400
$GET42  003602          1955#  1993
$GTSWR  017324          4235#  4652
$HARD  = 000054         2654   2675   2677*  5345#  5346
$HD    - 000000         1039
$HINUM  020610          1709*  234'   3055   4531   4539   4544*  4549#
$ICNT   001104          1279#
$INTAG  001135          1293#  4232*  4251   4271   4384
$ITEMB  001114          1283#  3186   3386*  3414*  3709*  3714*  3899*  3911
$LF     001214          1320#  3911   3982   4367   4377
$LKCSB  001226          1326#  2553*
$LKCSR  001224          1325#  2548   2554*
$LKS    001232          1328#  2558   2563*
$LLVEC  001234          1329#  2560
$LONUM  020612          1710*  4530   4537   4543*  4550#
$LPADR  001106          1280#
$LPERR  001110          1281#
$LPVEC  001230          1327#  2550
$LSTAD  034012          1764   6685*  6689#
$MAIL  = ****** U       1727   1736   3905   3935
$MEX   = 000001         5325#  5326
$MISPO= 000060          2656   2693   2695*  5347#  5348
$MNEW   020164          4238   4382#
$MODE   000003          5327#  5328
```

```
$MSWR   020153          4235    4380#
$NULL   001154          1301#   3953    3982
$OCNT   016506          4015*   4044*   4057#
$OMODE  016510          4010*   4014*   4019    4022*   4033*   4059#
$OPERC= 000034          2302*   2303*   2456    2458    2588    2604    3035    5340#   5341
$PACK = 000022          2072    2460    2489    2860*   5336#   5337
$PASS   001100          1276#
$PASSC= 000062          2622    2857*   3013    3017    3043*   5348#   5349
$PATTC= 000026          2054*   2055*   2280    2419*   2495*   5338#   5339
$POSIT= 000040          2155*   2156*   2306*   2307*   3006    3009    5341#   5342
$PREVA= 000030          2304    2320*   2469*   2503*   2504*   3627    3652    3656*   5061*   5062*   5339#   5340
$PREVO= 000024          2327    2329*   2435    2467*   5060*   5337#   5338
$QUES   001212          1318#   3911    3982    4289    4360    4377
$RAND   020512          2330    2390    2448    2455    4526#
$RDCHR  017576          4302#   4655
$RDDEC= ****** U        4657
$RDLIN  017666          4325#   4656
$RDOCT= ****** U        4657
$RDSZ = 000012          4318#
$READ = 000044          2151*   2152*   3000    3003    5342#   5343
$REG  = 000014          2959    5333#   5334
$REGAD  001160          1305#
$REG0   001162          1307#   3877    6369    6376    6378    6380    6381    6383    6385    6387    6388    6390    6392
                        6394    6397    6400    6402
$REG1   001164          1308#
$REG10  001202          1315#
$REG11  001204          1316#
$REG2   001166          1309#
$REG3   001170          1310#
$REG4   001172          1311#
$REG5   001174          1312#
$REG6   001176          1313#
$REG7   001200          1314#
$RESRE  020454          4502#   4658
$RETRY  015560          3600*   3617*   3690*   3809*   3814*   3820*   3821    3829#
$RPBA = 000076          2145    3285    3577    3582    3739    3758    3882    5354#   5355
$RPCA = 000100          2472    3883    5355#   5356
$RPCS = 000072          3880    5352#   5353
$RPDA = 000102          2470    2529    2535    2540    3637    3639    3884    5356#   5357
$RPDS = 000066          3268    3517    3532    3878    5350#   5351
$RPER = 000070          2465    3270    3506    3508    3511    3514    3520    3523    3526    3529    3535    3665    3818
                        3879    5351#   5352
$RPM1 - 000104          3885    5357#   5358
$RPWC = 000074          3287    3303    3741    3881    5353#   5354
$RPZRO  011642          2638    3071#
$RPZR4  011622          2647    2658    3067#
$RPZR5  011632          2624    3069#
$RPZR8  011612          2628    2633    3065#
$R2A  = ****** U        4659
$SAVRE  020416          4486#   4657
$SB2D   015564          2623    2646    2657    3843#
$SEC  = 000012          2050*   2354*   2401    2470*   2471*   2479*   3306    3588*   3675*   3790*   5062    5331#   5332
$SETUP= 000146          1684#   1704    1705    1707    1709    1711    1733    3868    3908    3910    4172    4177    4178
                        4208    4384
$SILO = 000110          2578    2585    2590    2959    3987    5359#
$SIZE   033716          1761    6668#
```

CROSS REFERENCE TABLE -- USER SYMBOLS


```
$SKI   = 000056          2655    2684    2686*   5346#   5347
$SOFT  = 000052          2653    2666    2668*   5344#   5345
$STUP  = 177777          1684#
$SUCA  = 000106          2153    2320    2469    3589    3650    3656    3676    3792    3876    3886    5358#   5359
$SVPC  = 000200          1255#   1260
$SWR   = 122000          1028#   1039    1043    1044    1045    1046    1317    1711    3860    3861    3862    3863    3864
                         3893    3900    3905    3909    3911
$TATUS= 000016           2126    3256    3450    3456    3460    3462    3464    3466    3468    3782    3798    3811    3816
                         5334#   5335
$TIME    007624          2010    2036    2612    2710#   3195
$TKB     001146          1298#   2776    4132    4153    4164    4189    4217    4244
$TKCNT   016736          4133#   4148*   4178    4195*   4309    4311*
$TKINT   016746          1732    1932    4148#   4169    4230
$TKQEN=  016745          4137#   4203    4314
$TKQIN   016740          4134#   4149*   4150    4201*   4202*   4203    4205*
$TKQOU   016742          4135#   4150*   4312    4313*   4314    4316*
$TKQSR   016744          4136#   4149    4205    4316
$TKS     001144          1297#   1818*   3458*   3500*   4132    4154*   4185*   4187    4193*   4215    4231*   4241    4253*
                         4273*
$TKSRV   017016          2516    2784    2785    4151    4164#
$TN    = 000001          1028#   1039
$TNPWR   020724          4572    4573    4593#
$TOTAL= 000050           2652    2702    2704*   2991    5343#   5344
$TPB     001152          1300#   3971*   3982
$TPFLG   001157          1304#   1743*   3929    3982
$TPS     001150          1299#   3969    3982
$TRAP    021010          1707    4623#
$TRAP2   021032          4634#   4645
$TRK   - 000013          2051*   2364*   2476    2478*   3305    3587*   3674*   3789*   5332#   5333
$TRP   - 000014          4638#   4647#   4648#   4649#   4650#   4651#   4652    4653#   4654    4655#   4656#   4657#   4658#
                         4659#
$TRPAD   021044          4628    4645#
$TSTNM   001102          1277#   3892    3911
$TTYIN   020122          4327    4328    4340    4358    43'2    4376#
$TYPBN-  ******  U       4651
$TYPDS   016512          4073#   4650
$TYPE    016044          3929#   4638    4646
$TYPEC   016214          3950    3957    3964    3969#   3970    4275
$TYPEX   016262          3975    3977    3980#
$TYPOC   016310          4013#   4647
$TYPON   016324          4012    4015#   4649
$TYPOS   016264          4008#   4648
$WRDL  = 000020          2056*   2175    2186    2188*   2189    2190*   2211    2220    2225    2236    2273    2278    2395*
                         2413    2486*   3281    3302    3590    3594*   3677    3681*   3740    5335#   5336
$WRDM  = 000004          2057*   2058*   2413*   2414*   2487*   2488*   3595*   3682*   5328#   5329
$OFILL   016507          4009*   4013*   4023    4058#
$40CAT   ******  U       3902
.        034014          1243#   1247#   1255    1256#   1258#   1260#   1264#   1274#   1321    1702    2086    2095    2115
                         2229    2270    2272    2274    2276    3509    3512    3515    3518    3521    3524    3527    3530
                         3533    3536    3911    3982    4127#   4'32    4136#   4137    4138#   4376#   4377    4384#   4613#
                         5370    5410    5450    5490    5530    5570    5610    5650    5688#   5757#   6366#   664?
```

```
A        2661#   2662    2671    2680    2689    2698
CKCHR    1028#   3147    3163
CKNUM    1028#   1804
COMMEN      1#   1168#
ENDCOM      1#   1168#
ERROR    1062#   1829    3272    3289    3422    3470    3475    3481    3486    3492    3498    3538    3543    3549    3555
         3561    3567    3584    3607    3610    3614    3620    3622    3629    3653    3659    3667    3669    3696    3700
         3825    5073
ESCAPE      1#   1168#
GETPRI      1#   1168#   4411    6674
GETSWR      1#   1028#   1168#   1733
MORETA   1268#   1322
MOVREG   1028#   3869
MULT        1#   1168#
NEWTST      1#   1168#
POP         1#   1168#   2963    3399    4114    4462    4507    4545
PUSH        1#   1168#   2948    3391    4073    4414    4487    4526
REPORT      1#   1168#
SCOPE    1063#
SETPRI      1#   1168#   4305
SETTRA   4638#   4647    4648    4649    4650    4652    4654    4655    4656    4657    4658
SETUP       1#   1168#   1697
SKIP        1#   1168#
SLASH       1#   1168#
SPACE    1168#
STARS       1#   1168#   1253    1270    1321    1347    1351    1378    1382    1394    1398    1684    1691    1969    1973
         2161    2165    2310    2314    2507    2511    3248    3252    3850    3856    3914    3985    4063    4131    4208
         4223    4294    4318    4388    4471    4517    4554    4617    4660    4664    5317    5321    5690    5694    5759
         5763    6056    6060    6662
SWRSU       1#   1168#   1711#
TRMTRP   4638#
TYPBIN      1#   1168#
TYPDEC      1#   1168#   3218
TYPNAM      1#   1168#
TYPNUM      1#   1168#
TYPOCS      1#   1168#
TYPOCT      1#   1168#   3215    4236
TYPTXT      1#   1168#
$$CMRE   1268#   1307    1308    1309    1310    1311    1312    1313    1314    1315    1316
$$CMTM   1268#
$$ESCA      1#   1168#
$$NEWT      1#   1168#
$$SET    4638#   4647    4648    4649    4650    4652    4654    4655    4656    4657    4658
$$SKIP      1#   1168#
.EQUAT      1#   1028#   1058
.HEADE      1#   1028#   1029
.KT11       1#
.SETUP      1#   1028#   1684
.SWRHI      1#   1028#   1039
.SWRLO   1028#   1046#   1047    1048    1049    1051    1054    1055    1056
.$ACT1      1#   1028#   1251
.$APTB      1#
.$APTH      1#
.$APTY      1#
.$ASTA      1#
.$CATC      1#   1028#   1241
```

```
.$CMTA      1#   1028#   1268
.$DB2D      1#   1028#   4552
.$DB2O      1#
.$DIV       1#   1028#   4386
.$EOP       1#
.$ERRO      1#   1028#   3854
.$ERRT      1#   1028#
.$MULT      1#
.$POWE      1#
.$RAND      1#   1028#   4515
.$RDDE      1#
.$RDOC      1#
.$READ      1#   1028#   4129
.$R2AZ      1#
.$SAVE      1#   1028#   4469
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#
.$SIZE      1#   1028#   6660
.$SUPR      1#
.$TRAP      1#   1028#   4615
.$TYPB      1#
.$TYPD      1#   1028#   4061
.$TYPE      1#   1028#   3912
.$TYPO      1#   1028#   3983
.$40CA      1#
.1170       1#


. ABS.  034014      000

 ERRORS DETECTED:  0

 CZRP1C.BIN,CZRP1C.LST/CRF/SOL/NL:TOC=CZRP1C.SML,CZRP1C.P11
 RUN-TIME:  15 24 1 SECONDS
 RUN-TIME RATIO: 208/42=4.9
 CORE USED:  34K  (67 PAGES)
```