RP11E

MULTI-DRIVE EXERCISER

MD-11-DZRP1-B

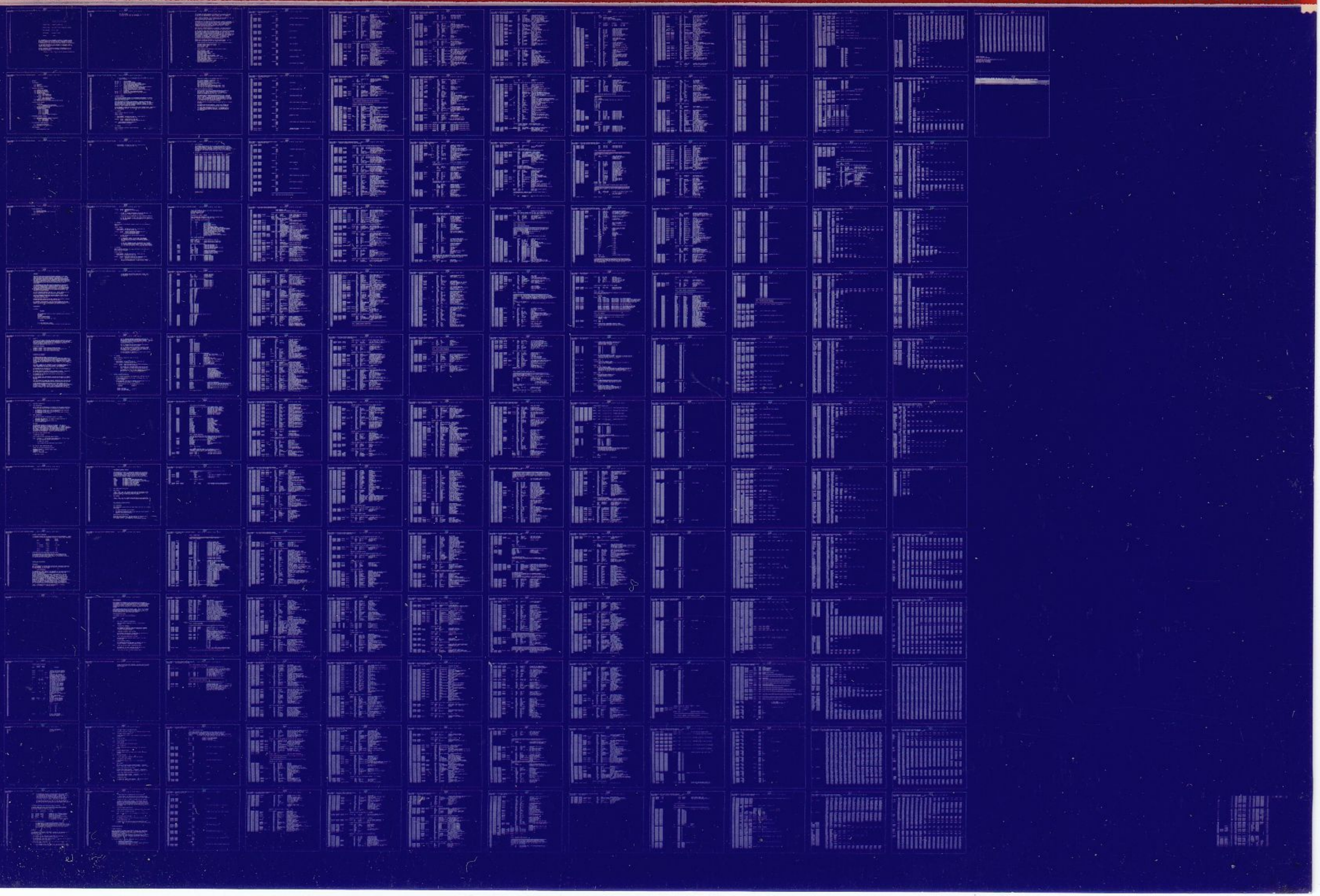
EP-DZRP1-B-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.



```
                                .REM    %
```

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

PRODUCT CODE: MAINDEC-11-DZRP1-B-D

PRODUCT NAME: RP11E MULTI-DRIVE EXERCISER PROGRAM

DATE CREATED: MAY 21, 1976

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: C. HESS

CONTENTS

```
101
```

1. ABSTRACT

THE RP11E MULTI-DRIVE EXERCISER PROGRAM 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 PERFORMING A DATA TRANSFER OR WRITE CHECK OPERATION). OPERATIONS AMONG THE DRIVES ARE OPTIMIZED SO THAT A HIGH SUBSYSTEM DATA TRANSFER RATE OR A HIGH 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 PATTERNS, AND DATA BUFFER SIZES ARE SELECTED RANDOMLY BY THE PROGRAM. ADDITIONALLY THE ADDRESSES (EG, CYLINDER, TRACK, AND SECTOR) FOR EACH OPERATION ARE SELECTED RANDOMLY.

2. REQUIREMENTS

2.1 EQUIPMENT

REQUIRED

PDP-11 PROCESSOR (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
MAINDEC-11-DZRPY - RP11E FUNCTIONAL LOGIC AND READ/WRITE TEST
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 ERROR 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

```
275
276                    (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'.)

333

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

AUTOCK	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> = 1 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>

```
511
512                          N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
513                              TERMINIATED BY A CARRIAGE RETURN <CR>.
```

EXAMPLE: DO<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: SO<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).

```
570                     IF THE BUFFER SIZE IS MUCH LESS THAN 1 TRACK, THE
571                     TIME REQUIRED TO WRITE A DATA PACK IS TOO GREAT TO
572                     BE PRACTICAL.
573
```

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.
RO<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 SEQUENTIAL.

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

```
630                             ?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.)

```
587
```

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

```
744
745
746
```

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 BECOME 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

%

```
                        .TITLE  MD-11-DZRP1-B, RP11E MULTI-DRIVE EXERCISER PROGRAM
                        ;*COPYRIGHT (C) 1975,1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY C. HESS/F. ROEMER
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE 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
```

```
                                R6=     %6              ;;GENERAL REGISTER
                                R7=     %7              ;;GENERAL REGISTER
                                .EQUIV  R6,SP           ;;STACK POINTER
                                .EQUIV  R7,PC           ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL DEFINITIONS
                                PR0=    0               ;;PRIORITY LEVEL 0
                                PR1=    40              ;;PRIORITY LEVEL 1
                                PR2=    100             ;;PRIORITY LEVEL 2
                                PR3=    140             ;;PRIORITY LEVEL 3
                                PR4=    200             ;;PRIORITY LEVEL 4
                                PR5=    240             ;;PRIORITY LEVEL 5
                                PR6=    300             ;;PRIORITY LEVEL 6
                                PR7=    340             ;;PRICRITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                                SW15=   100000
                                SW14=   40000
                                SW13=   20000
                                SW12=   10000
                                SW11=   4000
                                SW10=   2000
                                SW09=   1000
                                SW08=   400
                                SW07=   200
                                SW06=   100
                                SW05=   40
                                SW04=   20
                                SW03=   10
                                SW02=   4
                                SW01=   2
                                SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                                BIT15=  100000
                                BIT14=  40000
                                BIT13=  20000
                                BIT12=  10000
                                BIT11=  4000
                                BIT10=  2000
                                BIT09=  1000
                                BIT08=  400
                                BIT07=  200
                                BIT06=  100
                                BIT05=  40
                                BIT04=  20
```

```
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

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

                        ;RP11 REGISTER INDEX VALUES

000000                  RPDS=00                 ;DRIVE STATUS REGISTER
000002                  RPER=02                 ;ERROR REGISTER
000004                  RPCS=04                 ;CONTROL REGISTER
000006                  RPWC=06                 ;WORD COUNT REGISTER
000010                  RPBA=10                 ;BUFFER ADDRESS REGISTER
000012                  RPCA=12                 ;CYLINDER ADDRESS REGISTER
000014                  RPDA=14                 ;SECTOR/TRACK ADDRESS REGISTER
000016                  RPM1=16                 ;MAINTENANCE REGISTER #1
000020                  RPM2=20                 ;MAINTENANCE REGISTER #2
000022                  RPM3=22                 ;MAINTENANCE REGISTER #3
000024                  SUCA=24                 ;SELECTED UNIT CYLINDER ADDRESS REGISTER
000026                  SILO=26                 ;SILO REGISTER

                        ;RP11 OP CODE DEFINITIONS

000001                  CLEAR=1                 ;CLEAR THE CONTROLLER
000003                  WRTSEK=3                ;WRITE WITH IMPLIED SEEK AND HEAD SELECTION
000005                  ROSEK=5                 ;READ WITH IMPLIED SEEK AND HEAD SELECT
000007                  WRTCHK=7                ;WRITE CHECK WITH IMPLIIED SEEK AND HEAD SELECT
000011                  SEEK=11                 ;SEEK
000013                  WRITE=13                ;WRITE (NO IMPLIED SEEK OR HEAD SELECT)
000015                  HOMSEK=15               ;HOME SEEK
000017                  READ=17                 ;READ (NO IMPLIED SEEK OR HEAD SELECT)

                        ;RP11 REGISTER EQUATES
```

```
                                ;RPDS (DRIVE STATUS REGISTER)
                                ;ATTENTION BITS ARE REFERENCED BY BIT NUMBER (BIT00 - BIT07)
000400                          SUMP=400                ;SELECTED UNIT WRITE PROTECTED
001000                          SUFU=1000               ;SELECTED UNIT FILE UNSAFE
002000                          SUSU=2000               ;SELECTED UNIT SEEK UNDERWAY
004000                          SUSI=4000               ;SELECTED UNIT SEEK INCOMPLETE
010000                          HNF=10000               ;HEADER NOT FOUND
020000                          SURP03=20000            ;SELECTED UNIT IS AN RP03
040000                          SUOL=40000              ;SELECTED UNIT IS ONLINE
100000                          SURDY=100000            ;SELECTED UNIT IS READY

                                ;RPER (ERROR REGISTER)

000001                          DSKERR=1                ;DISK ERROR
000002                          EOP=2                   ;END OF PACK
000004                          NXME=4                  ;NON-EXISTENT MEMORY
000010                          WCE=10                  ;WRITE CHECK ERROR
000020                          TIMEE=20                ;TIMING ERROR
000040                          CSME=40                 ;CHECKSUM ERROR
000100                          WPE=100                 ;WORD PARITY ERROR
000200                          LPE=200                 ;LONGITUDINAL PARITY ERROR
000400                          MODERR=400              ;MODE ERROR
001000                          FMTE=1000               ;FORMAT ERROR
002000                          PROG=2000               ;PROGRAM ERROR
004000                          NXS=4000                ;NON-EXISTENT SECTOR
010000                          NXT=10000               ;NON-EXISTENT TRACK
020000                          NXC=20000               ;NON-EXSITENT CYLINDER
040000                          FUV=40000               ;FILE UNSAFE VIOLATION
100000                          WPV=100000              ;WRITE PROTECT VIOLATION

                                ;RPCS  (CONTROL REGISTER)

                                ;'GO' BIT AND FUNCTION BITS ARE LOADED TOGETHER AND ARE DEFINED ELSEWHERE.
                                ;EXTENDED MEMORY ADDRESS BITS ARE REFERED TO BY BIT NUMBER
000100                          IDE=100                 ;INTERRUPT ON DONE
000200                          RDY=200                 ;RP11 READY
                                ;DRIVE SELECT BITS ARE REFERED TO BY BIT NUMBER
004000                          HDR=4000                ;HEADER
010000                          MODE=10000              ;MODE
020000                          AIE=20000               ;ATTENTION INTERRUPT ENABLE
040000                          HE=40000                ;HARD ERROR
100000                          ERR=100000              ;ERROR

                                .SBTTL  TRAP CATCHER

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

                                .SBTTL  ACT11 HOOKS
```

```
1227
1228                                    ;;*****************************************************************
1229                                    ;HOOKS REQUIRED BY ACT11
1230            000200                          $SVPC=.                 ;SAVE PC
1231            000046                          .=46
1232  000046    003634                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1233            000052                          .=52
1234  000052    040000                          .WORD   40000           ;;2)SET LOC.52 TO 40000
1235            000200                          .=$SVPC                 ;; RESTORE PC
1236
1237                                    .SBTTL  STARTING ADDRESSES
1238
1239            000200                          .=200
1240
1241  000200    000137  002016                  JMP     START           ;START PROGRAM WITHOUT CHANGING PARAMETERS
1242  000204    000137  002024                  JMP     START1          ;START PROGRAM AND CHANGE PARAMETERS
```

```
                                .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
```

```
1299  001222  000240                RPPRIO: .WORD   5*32.          ;RP11 PRIORITY
1300  001224  172540                SLKCSR: .WORD   172540         ;ADDR OF KW11-P STATUS REGISTER
1301  001226  172542                SLKCSB: .WORD   172542         ;ADDR OF KW11-P COUNTER BUFFER
1302  001230  000104                SLPVEC: .WORD   104            ;ADDR OF KW11-P VECTOR
1303  001232  177546                SLKS:   .WORD   177546         ;ADDR OF KW11-L STATUS REGISTER
1304  001234  000100                SLLVEC: .WORD   100            ;ADDR OF KW11-L VECTOR
1305  001236  177777                CLKFLG: .WORD   -1             ;'0' IF A CLOCK IS AVAILABLE
1306  001240  000074                HZ:     .WORD   74             ;74 (8) IF 60 HZ SYSTEM; 62 (8) IF 50 HZ SYSTEM
1307  001242  000000                CNTRLC: .WORD   0              ;VECTOR ADDRESS FOR CONTROL C
1308  001244  000000                LSTAD:  .WORD   0              ;CONTAINS LAST MEMORY ADDRESS
1309  001246  001100                STACK1: .WORD   STACK          ;CONSTANT USED TO CHECK STACK POINTER
1310  001250  000000                STATIN: .WORD   0              ;ERROR RATE DATA DISPLAY INDICATOR
1311  001252  000000                PACK:   .WORD   0              ;'W' OR 'R' COMMAND INDICATOR
1312  001254  000003                FAIRNS: .WORD   3              ;'FAIRNESS' COUNT FOR BUFFER WAIT
1313  001256  000000                MAXCYL: .WORD   0              ;MAXIMUM CYLINDER ADDRESS ALLOWED FOR
1314                                                              ;ANY DRIVE TESTED.  (FILLED IN DURING STARTUP.)
1315  001260  001750                TIMOUT: .WORD   1000.          ;TIME (IN MS) ALLOWED FOR OPERATION
1316  001262  000000                CHNGPM: .WORD   0              ;IS NON-ZERO IF PARAMETERS TO BE CHANGED
1317  001264  000000                DRIVE:  .WORD   0              ;STORE DRIVE NUMBER HERE
1318  001266  000000                CYL.ER: .WORD   0              ;STORE CYLINDER ADDRESS HERE
1319  001270  000000                TRK.ER: .WORD   0              ;STORE TRACK ADDRESS HERE
1320  001272  000000                SEC.ER: .WORD   0              ;STORE SECTOR ADDRESS HERE
1321
1322                                ;;*****************************************************************
1323
1324                                .SBTTL  CONTROL PARAMETERS
1325
1326                                ;;*****************************************************************
1327
1328  001274  137100                ENDCON: .WORD   137100         ;6.25X10↑8 WORDS (10) (1X10↑10 BITS)
1329  001276  022500                        .WORD   22500          ;MSW
1330  001300  041100                ENDSEK: .WORD   041100         ;1 X 10↑6 SEEKS  (LSW)
1331  001302  000017                        .WORD   17             ;MSW
1332  001304  000001                PASCNT: .WORD   1              ;NUMBER OF PASSES TO END OF TEST
1333  001306  000000                BUFRSZ: .WORD   0              ;MAXIMUM DATA TRANFER SIZE IN WORDS
1334                                                              ;(FILLED BY PROGRAM AT STARTUP OR BY OPERATOR
1335                                                              ;DURING PARAMETER ENTRY DIALOG.)
1336  001310  000144                MAXER:  .WORD   100.           ;MAXIMUM ERRORS - 100(10)
1337  001312  000005                INTRVL: .WORD   5              ;LOW BYTE IS PERFORMANCE TYPEOUT INTERVAL
1338                                                              ;COUNTER.  UPPER BYTE IS VALUE.
1339  001314  000004                CMPLMT: .WORD   4              ;NUMBER OF COMPARE ERRORS TYPED OUT
1340  001316  000000                SEKMOD: .WORD   0              ;IF EQ 1, END OF PASS DETERMINED BY SEEK COUNT
1341  001320  000003                RATIO:  .WORD   3              ;READ/WRITE RATIO (RANGE 0 - 7)
1342                                                              ;0 - 15/1       (READ/WRITE)
1343                                                              ;1 - 7/1
1344                                                              ;2 - 6/2
1345                                                              ;3 - 5/3
1346                                                              ;4 - 4/4
1347                                                              ;5 - 3/5
1348                                                              ;6 - 2/6
1349                                                              ;7 - 1/7
1350  001322  000001                AUTOCK: .WORD   1              ;IF EQ 1, DO A WRITE CHECK AFTER EACH ORDER
1351                                                              ;IF EQ 0, SELECT WRITE CHECK ORDERS RANDOMLY
1352
1353                                ;;*****************************************************************
1354
```

```
                                .SBTTL  RP11E ADDRESS LIMIT VALUES

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

001324  000000                  LC:     .WORD   0               ;MAX CYLINDER [RANGE 0 - 312 OR 625 (OCTAL)]
                                                                ;THE PROGRAM WILL FILL THIS VALUE INITIALLY,
                                                                ;THE VALUE WILL DEPEND ON THE TYPE OF DRIVE
                                                                ;ON THE SYSTEM - RP02 OR RP03.
001326  000000                  FC:     .WORD   0               ;MINIMUM CYLINDER [RANGE 0 - (= LC]
001330  000023                  LT:     .WORD   23              ;MAX TRACK [RANGE 0 - 23 (OCTAL)]
001332  000000                  FT:     .WORD   0               ;MIN TRACK [RANGE 0 TO (= LT]
001334  000011                  LS:     .WORD   11              ;MAX SECTOR [RANGE 0 - 11 (OCTAL)]
001336  000000                  FS:     .WORD   0               ;MIN SECTOR [RANGE 0 TO (= LS]

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

                                .SBTTL  VALUES FOR FIRST OPERATION

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

001340  000010                  BEGPAT: .WORD   10              ;STARTING PATTERN CODE [RANGE 1 - 17 (OCTAL)]
001342  000005                  BEGCOD: .WORD   5               ;STARTING COMMAND CODE
                                                                ;PRELOADED TO PERFORM A READ WITH IMPLIED SEEK
001344  000400                  BEGSIZ: .WORD   400             ;STARTING RECORD SIZE [RANGE 4 - MAXMEM]
                                                                ;NOTE:  THE SIZE MUST BE AT LEAST 4.
                                                                ;IF THE SIZE IS GREATER THAN 1 SECTOR, THE
                                                                ;SIZE MUST ALLOW FOR OVERLAPPING 4
                                                                ;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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF SITEMB 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  027030                          EM1                     ;RP11 DIDN'T RESPOND TO ADDRESSING
001350  030352                          DH1
001352  032010                          DT1
001354  032264                          DF1

                                ;ERROR 2

001356  027072                          EM2                     ;INVALID RP11 COMMAND
001360  030360                          DH2
001362  032012                          DT2
001364  032270                          DF2

                                ;ERROR 3

001366  027117                          EM3                     ;UNDEFINED INTERRUPT FROM RP11
001370  030375                          DH3
001372  032016                          DT3
001374  032270                          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
```

```
1439  001424  000000                    0
1440
1441                                ;ERROR 7
1442
1443  001426  000000                    0                ;UNUSED
1444  001430  000000                    0
1445  001432  000000                    0
1446  001434  000000                    0
1447
1448                                ;ERROR 10
1449
1450  001436  027161                    EM10             ;CAN'T MATCH DATA READ WITH A PATTTERN
1451  001440  030412                    DH10
1452  001442  032026                    DT10
1453  001444  032274                    DF10
1454
1455                                ;ERROR 11
1456
1457  001446  000000                    0                ;SECOND LINE OF 'CAN'T MATCH DATA' MESSAGE
1458  001450  000000                    0
1459  001452  032022                    DT11
1460  001454  032270                    DF2
1461
1462                                ;ERROR 12
1463
1464  001456  027226                    EM12             ;DATA COMPARE ERRORS
1465  001460  030412                    DH10
1466  001462  032026                    DT10
1467  001464  032304                    DF12
1468
1469                                ;ERROR 13
1470
1471  001466  000000                    0                ;DATA COMPARE DETAIL LINE
1472  001470  000000                    0
1473  001472  032040                    DT13
1474  001474  032314                    DF13
1475
1476                                ;ERROR 14
1477
1478  001476  000000                    0                ;DATA COMPARE ERROR SUMMARY LINE
1479  001500  030571                    DH14
1480  001502  032046                    DT14
1481  001504  032320                    DF14
1482
1483                                ;ERROR 15
1484
1485  001506  027252                    EM15             ;'ERR' NOT SET BUT 'RPER' CONTENTS NOT ZERO
1486  001510  030617                    DH15
1487  001512  032050                    DT15
1488  001514  032324                    DF15
1489
1490                                ;ERROR 16
1491
1492  001516  027325                    EM16             ;WORD COUNT NOT ZERO OR BUFFER ADDRESS WRONG
1493  001520  030654                    DH16
1494  001522  032060                    DT16
```

```
001524  032330                          DF16
                                
                                ;ERROR 17
                                
001526  027401                          EM17            ;SOFTWARE TIMEOUT DURING POSITIONING
001530  030775                          DH17
001532  032076                          DT17
001534  032334                          DF17
                                
                                ;ERROR 20
                                
001536  027445                          EM20            ;SOFTWARE TIMEOUT DURING I/O
001540  030775                          DH17
001542  032076                          DT17
001544  032334                          DF17
                                
                                ;ERROR 21
                                
001546  027501                          EM21            ;DRIVE OFFLINE
001550  030775                          DH17
001552  032076                          DT17
001554  032334                          DF17
                                
                                ;ERROR 22
                                
001556  027517                          EM22            ;DRIVE UNSAFE
001560  030775                          DH17
001562  032076                          DT17
001564  032334                          DF17
                                
                                ;ERROR 23
                                
001566  027534                          EM23            ;SEEK INCOMPLETE
001570  031117                          DH23
001572  032124                          DT23
001574  032344                          DF23
                                
                                ;ERROR 24
                                
001576  027554                          EM24            ;UNIBUS TRANSFER ERROR
001600  031237                          DH24
001602  032144                          DT24
001604  032354                          DF24
                                
                                ;ERROR 25
                                
001606  027602                          EM25            ;CONTROLLER ERROR
001610  031304                          DH25
001612  032156                          DT25
001614  032324                          DF15
                                
                                ;ERROR 26
                                
001616  027623                          EM26            ;NON-EXISTENT DISK ADDRESS
001620  031341                          DH26
001622  032166                          DT26
```

```
1551  001624  032360                        DF26
1552
1553                                ;ERROR 27
1554
1555  001626  027655                        EM27            ;DATA ERROR
1556  001630  031416                        DH27
1557  001632  032202                        DT27
1558  001634  032364                        DF27
1559
1560                                ;ERROR 30
1561
1562  001636  000000                        0               ;SOFT ERROR REPORT LINE
1563  001640  031570                        DH30
1564  001642  000000                        0
1565  001644  000000                        0
1566
1567                                ;ERROR 31
1568
1569  001646  000000                        0               ;HARD ERROR REPORT LINE
1570  001650  031603                        DH31
1571  001652  000000                        0
1572  001654  000000                        0
1573
1574                                ;ERROR 32
1575
1576  001656  027670                        EM32            ;WRITE CHECK ERROR (NO DATA ERROR)
1577  001660  031416                        DH27
1578  001662  032202                        DT27
1579  001664  032364                        DF27
1580
1581                                ;ERROR 33
1582
1583  001666  027732                        EM33            ;WRITE CHECK ERROR (DATA ERROR BITS SET)
1584  001670  031416                        DH27
1585  001672  032202                        DT27
1586  001674  032364                        DF27
1587
1588                                ;ERROR 34
1589
1590  001676  030002                        EM34            ;HEADER NOT FOUND
1591  001700  031616                        DH34
1592  001702  032236                        DT34
1593  001704  032374                        DF34
1594
1595                                ;ERROR 35
1596
1597  001706  030023                        EM35            ;FORMAT ERROR WHILE SEARCHING FOR INITIAL SECTOR
1598  001710  031416                        DH27
1599  001712  032202                        DT27
1600  001714  032364                        DF27
1601
1602                                ;ERROR 36
1603
1604  001716  030103                        EM36            ; HEADER NOT FOUND, CYLINDER IN HEADER
1605                                                        ;DOES NOT COMPARE
1606  001720  031616                        DH34
```

```
001722  032236                  DT34
001724  032374                  DF34

                        ;ERROR 37

001726  030171                  EM37            ;DIFFERENT ERROR DURING RETRY
001730  030775                  DM17
001732  032076                  DT17
001734  032334                  DF17

                        ;ERROR 40

001736  000000                  0               ;UNUSED
001740  000000                  0
001742  000000                  0
001744  000000                  0

                        ;ERROR 41

001746  000000                  0               ;RETRY SUCESSFUL
001750  031745                  DM41
001752  000000                  0
001754  000000                  0

                        ;ERROR 42

001756  000000                  0               ;RETRY UNSUCESSFUL
001760  031765                  DM42
001762  000000                  0
001764  000000                  0

                        ;ERROR 43

001766  030226                  EM43            ;ERROR SIGNALED BUT NO ERROR BITS SET
001770  030775                  DM17
001772  032076                  DT17
001774  032334                  DF17

                        ;ERROR 44

001776  030267                  EM44            ;WRITE PROTECT VIOLATION
002000  030617                  DM15
002002  032050                  DT15
002004  032324                  DF15

                        ;ERROR 45

002006  030322                  EM45            ;FORMAT ERROR DURING I/O
002010  031416                  DM27
002012  032202                  DT27
002014  032364                  DF27

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

                        .SBTTL  SETUP AND INITIALIZATION ROUTINE
```

```
                                ;       START ADDRESS = 200
                                ;       START ADDRESS WITH PARAMETER CHANGES = 204
                                ;
                                ;;*****************************************************************
002016 005037 001262            START:  CLR     CHNGPM          ;CLEAR 'START FROM 204' INDICATOR
002022 000403                           BR      BEGIN           ;START THE PROGRAM
002024 012737 177777 001262     START1: MOV     #-1,CHNGPM      ;SET 'CHANGE PARAMETER' INDICATOR
002032 000005                   BEGIN:  RESET                   ;CLEAR EVERYTHING
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002034 012706 001100                    MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002040 005026                           CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002042 022706 001140                    CMP     #SWR,R6 ;;DONE?
002046 001374                           BNE     .-6             ;;LOOP BACK IF NO
002050 012706 001100                    MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
002054 012737 015624 000030             MOV     #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002062 012737 000340 000032             MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002070 012737 021020 000034             MOV     #TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002076 012737 000340 000036             MOV     #340,@#TRAPVEC+2;LEVEL 7
002104 012737 176543 020620             MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
002112 012737 123456 020622             MOV     #123456,$LONUM  ;;BOTH HIGH AND LOW WORDS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002120 013746 000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002124 012737 002160 000004             MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002132 012737 177570 001140             MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002140 012737 177570 001142             MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002146 022777 177777 176764             CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002154 001012                           BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002156 000403                           BR      65$             ;;BRANCH IF NO TIMEOUT
002160 012716 002166            64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002164 000002                           RTI
002166 012737 000176 001140     65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002174 012737 000174 001142             MOV     #DISPREG,DISPLAY
002202 012637 000004            66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002206 012737 000240 000032             MOV     #240,@#EMTVEC+2 ;CHANGE EMT PRIORITY TO 5
002214 012737 000240 000036             MOV     #240,@#TRAPVEC+2        ;CHANGE TRAP PRIORITY TO 5
002222 104401 033644            1$:     TYPE    ,TITLE          ;TYPE THE PROGRAM NAME AND MAINDEC NUMBER
002226 012737 001213 002224             MOV     #$CRLF,1$+2     ;CHANGE 'TITLE' TO CR-LF
002234 005037 177776                    CLR     PS              ;RESET PROCESSOR PRIORITY
002240 004737 016756                    JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
002244 005737 000042                    TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
002250 001006                           BNE     67$             ;;BRANCH IF YES
002252 023727 001140 000176             CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
002260 001005                           BNE     68$             ;;BRANCH IF NO
002262 104406                           GTSWR                   ;;GET SOFT-SWR SETTINGS
002264 000403                           BR      68$
002266 112737 000001 001134     67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
002274                          68$:
002274 005037 001250                    CLR     STATIN          ;CLEAR PERFORMANCE SUMMARY INDICATOR
002300 105037 001157                    CLRB    $TPFLG          ;SET TTY AVAILABILITY FLAG
```

```
1719  002304  012705  025460          MOV     #ASMLST,R5      ;START OF AREA TO CLEAR
1720  002310  005025          2$:    CLR     (R5)+           ;CLEAR
1721  002312  022705  025736          CMP     #BLKADR,R5      ;LOOK FOR END OF CLEAR AREA
1722  002316  001374                  BNE     2$              ;BR IF NOT FINISHED
1723  002320  013737  001240  010144  MOV     HZ,SIXTEE       ;1/60 TH OR 1/50 TH SECOND COUNTER VALUE
1724  002326  005437  010144          NEG     SIXTEE          ;CONVERT TO 2'S COMP
1725  002332  012737  030060  010130  MOV     #30060,HOUR     ;ASCII '00' TO HOUR COUNTER
1726  002340  012737  030060  010134  MOV     #30060,MINUTE   ;ASCII '00' TO MINUTES COUNTER
1727  002346  012737  030060  010140  MOV     #30060,SECOND   ;ASCII '00' TO SECONDS COUNTER
1728  002354  105037  001313          CLRB    INTRVL+1        ;CLEAR INTERVAL COUNTER
1729  002360  005037  177776          CLR     PSW             ;CLEAR PROCESSOR STATUS
1730  002364  005037  001252          CLR     PACK            ;'R' OR 'W' COMMAND INDICATOR
1731
1732                                  ;ROUTINE TO DETERMINE BUFFER AREA SIZE
1733
1734  002370  005227  177777  SIZMEM: INC     #-1             ;FIRST START ?
1735  002374  001013                  BNE     1$              ;BR IF NOT
1736  002376  004737  033742          JSR     PC,$SIZE        ;SEE HOW MUCH MEMORY ON SYSTEM
1737  002402  012737  000137  002370  MOV     #137,SIZMEM     ;CLEAR ENTRY TO '$SIZE'
1738  002410  012737  002424  002372  MOV     #1$,SIZMEM+2    ;SAME
1739  002416  013737  034036  001244  MOV     $LSTAD,LSTAD    ;SAVE THE HIGH MEMORY ADDRESS
1740  002424  012737  000001  025614  1$:    MOV     #1,BUFTBL       ;LOAD NUMBER OF BUFFERS
1741  002432  012737  033644  025616  MOV     #ENDPGM,BUFTBL+2  ;STARTING ADDRESS OF BUFFER
1742  002440  013737  001244  025620  MOV     LSTAD,BUFTBL+4  ;LAST ADDR TO BUFFER ALLOCATION TABLE
1743  002446  162737  033644  025620  SUB     #ENDPGM,BUFTBL+4  ;SUBTRACT PROGRAM SPACE
1744  002454  000241                  CLC                     ;CLEAR THE 'C' BIT
1745  002456  006037  025620          ROR     BUFTBL+4        ;CONVERT TO WORD COUNT
1746  002462  022737  037776  001244  CMP     #37776,LSTAD    ;ONLY 8K ON THE SYSTEM ?
1747  002470  103003                  BHIS    2$              ;BR IF 8K OR LESS
1748  002472  105737  000041          TSTB    41              ;SEE WHO LOADED THE PROGRAM
1749  002476  001004                  BNE     3$              ;BR IF LOADED BY 'XXDP'
1750  002500  162737  000144  025620  2$:    SUB     #100.,BUFTBL+4  ;SUBTRACT 'ABS' LOADER SIZE
1751  002506  000403                  BR      4$              ;CONTINUE WITH BUFFER SPACE SETUP
1752  002510  162737  002734  025620  3$:    SUB     #1500.,BUFTBL+4  ;SUBTRACT 'XXDP' LOADER SIZE
1753  002516  005737  001306  4$:     TST     BUFRSZ          ;VALUE IN 'BUFRSZ' ?
1754  002522  001012                  BNE     5$              ;BR IF VALUE IS
1755  002524  012737  005400  001306  MOV     #2816.,BUFRSZ   ;ASSUME FULL TRACK + 1 SEC MAXIMUM
1756  002532  023737  001306  025620  CMP     BUFRSZ,BUFTBL+4 ;IS THAT TOO LARGE ?
1757  002540  101403                  BLOS    5$              ;BR IF NOT
1758  002542  013737  025620  001306  MOV     BUFTBL+4,BUFRSZ ;USE MAX AVAIL MEMORY AS MAX BUFFER SIZE
1759  002550  005737  001262  5$:     TST     CHNGPM          ;CHANGE PARAMETERS ?
1760  002554  001455                  BEQ     CKADR           ;BR IF NOT
1761
1762                                  ;PARAMETER ENTRY ROUTINE
1763
1764  002556  012737  002662  001242  ENTPAR: MOV     #ENTPR1,CNTRLC  ;CONTROL C RETURN ADDRESS
1765  002564  104401  033374          ENTPR:  TYPE    ,ASKPAR         ;'ENTER PARAMETERS'
1766  002570  012704  033276          MOV     #PARLST,R4      ;ENTRY PARAMETER LIST TO R4
1767  002574  012437  002612          1$:     MOV     (R4)+,4$        ;ADDRESS OF PARAMETER NAME
1768  002600  001435                  BEQ     ENTPR2          ;BR IF AT END OF TABLE
1769  002602  000402                  BR      3$              ;GET AROUND THE CR-LF
1770  002604  104401  001213          2$:     TYPE    ,$CRLF          ;CR-LF
1771  002610  104401                  3$:     TYPE                    ;TYPE THE PARAMETER NAME
1772  002612  000000                  4$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
1773  002614  011405                  MOV     (R4),R5         ;ADDRESS OF PARAMETER LOCATION
1774  002616  011546                  MOV     (R5),-(SP)      ;CONTENTS OF PARAMETER LOCATION
```

```
                                        TYPOC                   ;TYPE THE CURRENT VALUE
                    032501              TYPE    ,SPACE1         ;1 SPACE
                                        RDLIN                   ;READ THE KEYBOARD
                                        MOV     (SP)+,R1        ;BUFFER ADDRESS
                    012020              JSR     R0,CK.NUM       ;CHECK THE NUMBER
                                        2$                      ;ILLEGAL INPUT
                                        5$                      ;TERMINATED WITH A 5$,
                                        7$                      ;TERMINATED WITH A ',
                                        6$                      ;NO ENTRY - TERMINATED WITH A 5$
                                        ENTPR2                  ;NO ENTRY - TERMINATED WITH A ',
                                5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
                                6$:     TST     (R4)+           ;INCREMENT THE PARAMETER POINTER
                                        BR      1$              ;GET MORE PARAMETERS
                                7$:     MOV     R2,(R5)         ;NEW PARAMETER VALUE
                                        BR      ENTPR2          ;CONTINUE
                    001100      ENTPR1: MOV     #STACK,SP       ;RESET THE STACK POINTER
                    177776              CLR     PSW             ;CLEAR THE STATUS WORD
                                        BR      ENTPR           ;START WITH PARAMETER ENTRY AGAIN
                    176244      ENTPR2: CLR     @STKS           ;CLEAR THE KEYBOARD INTERRUPT
                    177776              CLR     PSW             ;CLEAR PROCESSOR STATUS
                    001100              MOV     #STACK,SP       ;RESTORE STACK POINTER FOR RESTART

                                ;CHECK THE PARAMETERS FOR VALIDITY

                    002732      000004  CKADR:  MOV     #1$,ERRVEC      ;CHANGE TIMEOUT RETURN
                    176274              TST     @RPADR          ;SEE IF RP11 RESPONDS
                    000006      000004          MOV     #ERRVEC+2,ERRVEC  ;RESET THE TIMEOUT VECTOR
                                        BR      2$              ;CONTINUE WITH CHECKING
                    000006      000004  1$:     MOV     #ERRVEC+2,ERRVEC  ;RESET TIMEOUT VECTOR
                                        ERROR   1               ;REPORT RP11 DIDN'T RESPOND
                    000042              MOV     @#42,R0         ;'CHAIN MODE ' OR 'ACT11' ?
                                        BEQ     ENTPAR          ;IF NOT - GO TO ENTER PARAMETER ROUTINE
                    003634              JMP     SENDAD          ;RETURN TO MONITOR
                    021314      2$:     JSR     PC,RPINIT       ;INITIALIZE THE RP11 HANDLER
                    177777      021132  MOV     #-1,SAVEFG      ;SET THE 'SAVE REGISTER' FLAG
                    000312      001256  MOV     #202.,MAXCYL    ;START CYLINDER LIMIT AT RP02 LIMIT
                    021122              TST     DRVTYP          ;SEE WHICH DRIVES (RP02 OR RP03) ARE
                                                                ;ON SYSTEM
                                        BEQ     3$              ;BR IF RP02'S
                    000625      001256  MOV     #405.,MAXCYL    ;CYLINDER LIMIT FOR RP03'S
                    001324      3$:     TST     LC              ;VALUE ENTERED IN 'LC' ?
                                        BNE     4$              ;BR IF ONE WAS
                    001256      001324  MOV     MAXCYL,LC       ;USE MAXIMUM VALUE
                    001262      4$:     CLR     CHNGPM          ;CLEAR CHANGE PARAMETERS INDICATOR
                    001306      025620  CMP     BUFRSZ,BUFTBL+4 ;REQUESTED TRANSFER SIZE TOO LARGE ?
                                        BHI     5$              ;BR IF REQUESTED MAXIMUM TOO LARGE
                    000004      001306  CMP     #4,BUFRSZ       ;SEE IF MAXIMUM TRANSFER SIZE TOO SMALL
                                        BLOS    6$              ;BR IF NOT TOO SMALL
                    033501      5$:     MOV     #PAR1,-(SP)     ;ADDR OF PARAMETER NAME
                    003256              JMP     BADPAR          ;REPORT THE PARAMETER ERROR
                    001306      001344 6$:     CMP     BUFRSZ,BEGSIZ   ;IS MAX RECORD LENGTH SMALLER THAN 1 SECTOR ?
                                        BHIS    CKPAR           ;BR IF NOT
                    001306      001344  MOV     BUFRSZ,BEGSIZ   ;USE MAX RECORD SIZE AS STARTING SIZE
                    001256      001324  CKPAR:  CMP     MAXCYL,LC       ;MAXIMUM CYLINDER ADDR TOO LARGE ?
                                        BHIS    1$              ;BR IF NOT
                    033531              MOV     #PAR4,-(SP)     ;ADDR OF PARAMETER NAME
```

```
 1831  003112  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1832  003116  023737  001324  001326  1$:       CMP     LC,FC           ;MAX CYLINDER LARGER THAN MIN CYLINDER
 1833  003124  103004                            BHIS    2$              ;BR IF IT IS
 1834  003126  012746  033535                    MOV     #PAR5,-(SP)     ;ADDR OF PARAMETER NAME
 1835  003132  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1836  003136  022737  000023  001330  2$:       CMP     #19.,LT         ;MAXIMUM TRACK ADDRESS TOO LARGE ?
 1837  003144  103004                            BHIS    3$              ;BR IF NOT
 1838  003146  012746  033541                    MOV     #PAR6,-(SP)     ;ADDRESS OF PARAMETER NAME
 1839  003152  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1840  003156  023737  001330  001332  3$:       CMP     LT,FT           ;SEE IF MAX TRACK EQ OR LARGER THAN MIN TRACK
 1841  003164  103004                            BHIS    4$              ;BR IF IT IS
 1842  003166  012746  033545                    MOV     #PAR7,-(SP)     ;ADDRESS OF PARAMETER NAME
 1843  003172  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1844  003176  022737  000011  001334  4$:       CMP     #9.,LS          ;SEE IF MAX SECTOR TOO LARGE ?
 1845  003204  103004                            BHIS    5$              ;BR IF MAX SECTOR OK
 1846  003206  012746  033551                    MOV     #PAR8,-(SP)     ;ADDRESS OF PARAMETER NAME
 1847  003212  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1848  003216  023737  001334  001336  5$:       CMP     LS,FS           ;MINIMUM SECTOR ADDRESS TOO LARGE
 1849  003224  103004                            BHIS    6$              ;BR IF NOT
 1850  003226  012746  033555                    MOV     #PAR9,-(SP)     ;ADDRESS OF PARAMETER NAME
 1851  003232  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1852  003236  022737  000007  001320  6$:       CMP     #7,RATIO        ;SEE IF R/W RATIO TOO LARGE
 1853  003244  103016                            BHIS    SETVEC          ;BR IF NOT
 1854  003246  012746  033607                    MOV     #PAR14,-(SP)    ;ADDRESS OF PARAMETER NAME
 1855  003252  000137  003256                    JMP     BADPAR          ;REPORT PARAMETER ERROR
 1856
 1857                                    ;REPORT THE BAD PARAMETER
 1858
 1859  003256  012637  003270            BADPAR: MOV     (SP)+,1$        ;MOVE PARAMETER NAME
 1860  003262  104401  033421                    TYPE    ,NGPAR          ;'BAD PARAMETER'
 1861  003266  104401                            TYPE
 1862  003270  000000                    1$:     .WORD   0               ;PARAMETER NAME
 1863  003272  104401  033451                    TYPE    ,REPAR          ;'REENTER PARAMETERS
 1864  003276  000137  002556                    JMP     ENTPAR          ;GO THROUGH PARAMETER ENTRY ROUTINE AGAIN
 1865
 1866                                    ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
 1867                                    ;       PROGRAM WILL USE
 1868
 1869  003302  005227  177777            SETVEC: INC     #-1             ;FIRST START ?
 1870  003306  001404                            BEQ     9$              ;BR IF IT IS
 1871  003310  032777  000200  175622            BIT     #SW07,@SWR      ;SWITCH 6 SET ?
 1872  003316  001064                            BNE     6$              ;BR IF SET
 1873  003320  005004                    9$:     CLR     R4              ;DRIVE TABLE POINTER
 1874  003322  012703  000010                    MOV     #8.,R3          ;ITERATION COUNTER
 1875  003326  104401  001213                    TYPE    ,$CRLF          ;CR-LF
 1876  003332  104401  032625                    TYPE    ,SYSTAT         ;TYPE STATUS HEADING
 1877  003336  010446                    1$:     MOV     R4,-(SP)        ;DRIVE NUMBER FOR TYPEOUT
 1878  003340  104403                            TYPOS                   ;TYPE THE DRIVE NUMBER
 1879  003342      001      000                   .BYTE   1,0
 1880  003344  104401  032476                    TYPE    ,SPACE4         ;SPACES
 1881  003350  105764  021112                    TSTB    DRVSTA(R4)      ;SEE WHAT DRIVE'S STATUS IS
 1882  003354  001405                            BEQ     2$              ;BR IF OFFLINE OR NOT PRESENT
 1883  003356  100413                            BMI     3$              ;BR IF UNSAFE
 1884  003360  012737  032535  003442            MOV     #UNTON,6$       ;MESSAGE ADDRESS
 1885  003366  000412                            BR      4$              ;DETERMINE DRIVE TYPE MESSAGE
 1886  003370  012737  032524  003442  2$:       MOV     #UNTOFF,6$      ;OFFLINE MESSAGE ADDRESS
```

```
 1887  003376  012737  032501  003452          MOV     #SPACE1,7$      ;DUMMY MESSAGE
 1888  003404  000415                          BR      5$              ;PRINT IT
 1889  003406  012737  032615  003442  3$:     MOV     #NOTSAF,6$      ;UNIT UNSAFE
 1890  003414  012737  032503  003452  4$:     MOV     #RP02,7$        ;ASSUME RP02
 1891  003422  136437  021304  021122          BITB    ATABIT(R4),DRVTYP  ;SEE WHICH DRIVE TYPE
 1892  003430  001403                          BEQ     5$              ;BR IF RP02
 1893  003432  012737  032510  003452          MOV     #RP03,7$        ;'RP03'
 1894  003440  104401                  5$:     TYPE                    ;TYPE THE IDENTIFYING MESSAGE
 1895  003442  000000                  6$:     .WORD   0               ;MESSAGE ADDRESS HERE
 1896  003444  104401  032476                  TYPE    ,SPACE4         ;SPACES
 1897  003450  104401                          TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
 1898  003452  000000                  7$:     .WORD   0               ;DRIVE TYPE MESSAGE GOES HERE
 1899  003454  104401  001213                  TYPE    ,$CRLF          ;CR-LF
 1900  003460  005303                          DEC     R3              ;DECREMENT THE ITERATION COUNTER
 1901  003462  001402                          BEQ     8$              ;BR IF END
 1902  003464  005204                          INC     R4              ;INCREMENT TABLE POINTER
 1903  003466  000723                          BR      1$              ;CONTINUE
 1904  003470  104401  033212                  TYPE    ,INTDON         ;TYPE 'INITIALIZE COMPLETE'
 1905  003474  004737  006762                  JSR     PC,CKCLK        ;START CLOCK IF ONE ON SYSTEM
 1906  003500  012737  000200  177776          MOV     #PR4,PSW        ;SET PRIORITY TO KEYBOARD LEVEL
 1907  003506  004737  016756                  JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD INTERRUPT HANDLER
 1908  003512  012737  006640  001242          MOV     #DORTI,CNTRLC   ;ADDRESS IF 'CONTROL C'
 1909  003520  012737  010146  000060          MOV     #KSR,TKVEC      ;CHANGE KEYBOARD VECTOR ADDRESS
 1910  003526  005037  177776                  CLR     PSW             ;LOWER PRIORITY
 1911
 1912                                  ;SET UP IF 'XXDP' OR 'ACT11' OPERATION
 1913
 1914  003532  005737  000042          MONTR:  TST     42              ;'XXDP' CHAIN MODE OR 'ACT11' AUTO ACCEPT
 1915  003536  001417                          BEQ     2$              ;BR IF NEITHER
 1916  003540  012703  000010                  MOV     #8.,R3          ;DRIVE COUNTER
 1917  003544  005004                          CLR     R4              ;DRIVE POINTER
 1918  003546  105737  000041                  TSTB    41              ;SEE WHICH - 'XXDP' OR 'ACT11' ?
 1919  003552  001407                          BEQ     1$              ;BR IF 'ACT11' CONTROL
 1920  003554  122737  000007  000041          CMPB    #7,41           ;IS RP02 OR RP03 THE LOADING DEVICE ?
 1921  003562  001003                          BNE     1$              ;BR IF NOT
 1922  003564  005204                          INC     R4              ;SETUP TO BYPASS DRIVE 0
 1923  003566  012703  000007                  MOV     #7,R3           ;DRIVES TO TRY TO ASSIGN
 1924  003572  004737  010502          1$:     JSR     PC,ASGN2        ;ASSIGN ALL AVAILABLE DRIVES (EXCEPT DRIVE
 1925                                                                  ;0 IF LOADED FROM AN RP02 OR RP03)
 1926  003576  000137  004046          2$:     JMP     MAIN1           ;START THE PROGRAM
 1927
 1928                                  ;'XXDP' OR 'ACT11' END OF TEST ROUTINE
 1929
 1930  003602  013700  000042          $GET42: MOV     42,R0           ;MONITOR ADDRESS
 1931  003606  001002                          BNE     1$              ;BR IF MONITOR ADDRESS
 1932  003610  000137  004046                  JMP     MAIN1           ;NONE, CONTINUE
 1933  003614  022700  003634          1$:     CMP     #$ENDAD,R0      ;IS MONITOR 'ACT11'
 1934  003620  001005                          BNE     $ENDAD          ;NO, BRANCH
 1935  003622  022760  177777  000002          CMP     #-1,2(R0)       ;YES, IF THIS LAST PASS
 1936  003630  001005                          BNE     $DOAGN          ;NO, MAKE ANOTHER PASS
 1937  003632  000005                          RESET                   ;CLEAR EVERYTHING
 1938  003634  004710                  $ENDAD: JSR     PC,(R0)         ;GO TO THE MONITOR
 1939  003636  000240                          NOP                     ;SAVE ROOM
 1940  003640  000240                          NOP                     ;FOR
 1941  003642  000240                          NOP                     ;ACT11
 1942  003644  000137  002016          $DOAGN: JMP     START           ;START AGAIN
```

```
1943
1944                                    ;;*****************************************************************
1945
1946                                    .SBTTL  MAIN PROGRAM
1947
1948                                    ;;*****************************************************************
1949
1950  003650  005737  001250           MAIN:   TST     STATIN          ;TYPE ERROR RATE STATISTICS ?
1951  003654  001410                           BEQ     MAIN0           ;BR IF NOT
1952  003656  032777  000100  175254           BIT     #SW06,@SWR      ;CHECK SWITCH 6
1953  003664  001004                           BNE     MAIN0           ;BR IF SET
1954  003666  005037  001250                   CLR     STATIN          ;CLEAR THE INDICATOR
1955  003672  004737  007126                   JSR     PC,STATPR       ;TYPE THE STATISTICS
1956
1957                                    ;LOOK FOR DRIVES TO BE DEASSIGNED
1958
1959  003676  012703  000010           MAIN0:  MOV     #8.,R3          ;UNIT COUNTER
1960  003702  012705  025504                   MOV     #DUNIT,R5       ;ADDRESS OF 'DROP UNIT' TABLE
1961  003706  005715                   1$:     TST     (R5)            ;SEE IF ENTRY AT PRESENT POSITION
1962  003710  001011                           BNE     3$              ;BR IF THERE IS ONE
1963  003712  062705  000002           2$:     ADD     #2,R5           ;INCREMENT TO NEXT TABLE POSITION
1964  003716  005303                           DEC     R3              ;DECREMENT UNIT COUNTER
1965  003720  001372                           BNE     1$              ;BR IF MORE TO CHECK
1966  003722  005737  025460                   TST     ASNLST          ;ANY DRIVES STILL RUNNING ?
1967  003726  001047                           BNE     MAIN1           ;BR IF ANY ARE - CHECK ASSIGN QUEUE
1968  003730  000137  003602                   JMP     $GET42          ;CHECK FOR MONITOR RETURN
1969  003734  012701  025550           3$:     MOV     #AVAIL,R1       ;ADDRESS OF 'AVAILABLE' UNITS TABLE
1970  003740  005711                   4$:     TST     (R1)            ;SEE IF AT END OF TABLE
1971  003742  001405                           BEQ     5$              ;BR IF AT END: GO CHECK 'WAIT' TABLE
1972  003744  021115                           CMP     (R1),(R5)       ;IS UNIT IN 'AVAIL' THE ONE TO BE DROPPED
1973  003746  001414                           BEQ     7$              ;BR IF YES
1974  003750  062701  000002                   ADD     #2,R1           ;INCREMENT 'AVAIL' TABLE ADDRESS
1975  003754  000771                           BR      4$              ;CONTINUE LOOKING
1976  003756  012701  025572           5$:     MOV     #WAIT,R1        ;MOVE THE ADDRESS OF THE BUFFER WAIT TABLE
1977  003762  005711                   6$:     TST     (R1)            ;AT THE END OF THE 'WAIT' TABLE ?
1978  003764  001752                           BEQ     2$              ;BR IF YES: SEE IF ANY MORE 'DROP' REQUESTS
1979  003766  021115                           CMP     (R1),(R5)       ;UNIT IN THE 'WAIT' TABLE ?
1980  003770  001403                           BEQ     7$              ;BR IF IT IS
1981  003772  062701  000002                   ADD     #2,R1           ;INCREMENT 'WAIT' TABLE ADDRESS
1982  003776  000771                           BR      6$              ;CONTINUE LOOK THROUGH THE 'WAIT' TABLE
1983  004000  011100           7$:     MOV     (R1),R0         ;PUT THE UNIT'S BLOCK ADDRESS IN R0
1984  004002  104401  001213                   TYPE    ,$CRLF          ;CR-LF
1985  004006  004737  007634                   JSR     PC,$TIME        ;TYPE THE TIME
1986  004012  104401  032515                   TYPE    ,UNTMSG         ;'DRIVE'
1987  004016  111046                           MOVB    (R0),-(SP)      ;DRIVE NUMBER
1988  004020  104403                           TYPOS                   ;TYPE IT
1989  004022     001     000                   .BYTE   1,0
1990  004024  104401  033067                   TYPE    ,DEASSG         ;'DEASSIGNED'
1991  004030  004737  007240                   JSR     PC,TYPEST       ;TYPE THE DRIVE'S PERFORMANCE SUMMARY
1992  004034  005015                           CLR     (R5)            ;CLEAR THE 'DROP UNIT' TABLE ENTRY
1993  004036  005011                           CLR     (R1)            ;REMOVE THE UNIT FROM THE 'AVAIL' OR 'WAIT' TABLE
1994  004040  004737  006650                   JSR     PC,CMPRES       ;COMPRESS THE RESPECTIVE TABLE
1995  004044  000722                           BR      2$              ;SEE IF ANY MORE UNITS
1996
1997                                    ;LOOK FOR DRIVES TO BE ASSIGNED
1998
```

```
 1999  004046  012703  000010          MAIN1:  MOV     #8.,R3          ;UNIT COUNT
 2000  004052  005002                          CLR     R2              ;'AVAIL' INDEX
 2001  004054  005004                          CLR     R4              ;ASSIGN LIST INDEX
 2002  004056  005005                          CLR     R5              ;NEW UNIT INDEX
 2003  004060  005765  025526          1$:     TST     NEWUNT(R5)      ;NEW UNIT IN THIS POSITION
 2004  004064  001006                          BNE     3$              ;BR IF THERE IS
 2005  004066  062705  000002          2$:     ADD     #2,R5           ;INCREMENT R5
 2006  004072  005204                          INC     R4              ;INCREMENT ASSIGN INDEX
 2007  004074  005303                          DEC     R3              ;DECREMENT UNIT COUNT
 2008  004076  001370                          BNE     1$              ;BR IF MORE UNITS
 2009  004100  000472                          BR      MAIN2           ;START OPERATIONS FOR THE AVAILABLE UNITS
 2010  004102  104401  001213          3$:     TYPE    ,$CRLF          ;CR-LF
 2011  004106  004737  007634                  JSR     PC,$TIME        ;TYPE THE TIME
 2012  004112  104401  032515                  TYPE    ,UNTMSG         ;'DRIVE'
 2013  004116  010446                          MOV     R4,-(SP)        ;DRIVE NUMBER
 2014  004120  104403                          TYPOS                   ;TYPE IT
 2015  004122     001     000                  .BYTE   1,0
 2016  004124  104401  033131                  TYPE    ,ASGND          ;'ASSIGNED'
 2017  004130  005762  025550          4$:     TST     AVAIL(R2)       ;AT END OF AVAILABLE TABLE
 2018  004134  001403                          BEQ     5$              ;BR IF YES
 2019  004136  062702  000002                  ADD     #2,R2           ;INCREMENT AVAILABLE TABLE INDEX
 2020  004142  000772                          BR      4$              ;CONTINUE LOOKING FOR END OF TABLE
 2021  004144  016562  025526  025550  5$:     MOV     NEWUNT(R5),AVAIL(R2)  ;MOVE ADDR OF UNIT INTO AVAIL LST
 2022  004152  005065  025526                  CLR     NEWUNT(R5)      ;TAKE UNIT OUT OF NEW UNIT TABLE
 2023  004156  156437  021304  025460          BISB    ATABIT(R4),ASNLST  ;SET UNIT ASSIGNED INDICATOR
 2024  004164  016200  025550                  MOV     AVAIL(R2),R0    ;PUT STARTING ADDRESS OF BLOCK IN R0
 2025  004170  113760  001336  000012          MOVB    FS,$SEC(R0)  ;INITIAL SECTOR VALUE
 2026  004176  113760  001332  000013          MOVB    FT,$TRK(R0)  ;INITIAL TRACK VALUE
 2027  004204  013760  001326  000010          MOV     FC,$CYL(R0)  ;INITIAL CYLINDER VALUE
 2028  004212  113760  001342  000002          MOVB    BEGCOD,$COMND(R0)  ;INITIAL COMMAND CODE
 2029  004220  113760  001340  000026          MOVB    BEGPAT,$PATTC(R0)  ;PATTERN CODE
 2030  004226  106360  000026                  ASLB    $PATTC(R0)      ;CONVERT CODE TO A TABLE INDEX
 2031  004232  013760  001344  000020          MOV     BEGSIZ,$WRDL(R0)  ;BEGINNING RECORD SIZE
 2032  004240  013760  001344  000004          MOV     BEGSIZ,$WRDM(R0)  ;VALUE FOR DATA TRANSFER
 2033  004246  005460  000004                  NEG     $WRDM(R0)       ;MAKE IT INTO 2'S COMPLEMENT
 2034  004252  013760  001254  000064          MOV     FAIRNS,$FAIR(R0)  ;LOAD FAIRNESS COUNT
 2035  004260  062702  000002                  ADD     #2,R2           ;INCREMENT 'AVAILABLE' TABLE POINTER
 2036  004264  000700                          BR      2$              ;LOOK FOR MORE UNITS
 2037
 2038                          ;GET PARAMETERS, BUFFER SPACE, AND START ORDERS FOR DRIVES IN
 2039                          ;       THE 'AVAILABLE' QUEUE
 2040
 2041  004266  005737  025572          MAIN2:  TST     WAIT            ;OUTSTANDING BUFFER REQUESTS
 2042  004272  001065                          BNE     MAIN3           ;BR IF THERE ARE
 2043  004274  005002                          CLR     R2              ;CLEAR 'AVAIL' TABLE POINTER
 2044  004276  005762  025550          1$:     TST     AVAIL(R2)       ;ANY UNITS WAITING FOR PARAMETERS
 2045  004302  001513                          BEQ     IDLE            ;BRANCH IF NONE
 2046  004304  016200  025550                  MOV     AVAIL(R2),R0    ;CONTROL BLOCK ADDR IN R0
 2047  004310  026037  000064  001254          CMP     $FAIR(R0),FAIRNS  ;HAS ENTRY BEEN IN THE QUEUE ?
 2048  004316  001010                          BNE     3$              ;BR IF IT HAS
 2049  004320  105760  000022                  TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR THE DRIVE ?
 2050  004324  001403                          BEQ     2$              ;BR IF NOT
 2051  004326  004737  006332                  JSR     PC,WRTPK        ;GET DATA PACK PARAMETERS
 2052  004332  000402                          BR      3$              ;GET THE BUFFER
 2053  004334  004737  005512          2$:     JSR     PC,GETPAR       ;LOAD NEW PARAMETERS
 2054  004340  005046                  3$:     CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
```

```
2055  004342  004737  004720                  JSR     PC,GETBUF       ;GET BUFFER
2056  004346  012660  000006                  MOV     (SP)+,$BUF(R0)  ;MOVE BUFFER ADDR TO DPB
2057  004352  001414                          BEQ     5$              ;BR IF '0' ADDR (NO BUFFER)
2058  004354  004737  005304                  JSR     PC,FILBUF       ;FILL THE BUFFER
2059  004360  005060  000064                  CLR     $FAIR(R0)       ;CLEAR THE 'FAIRNESS' COUNT
2060  004364  004737  005434                  JSR     PC,GODRIV       ;PUT CURRENT DPB IN DRIVER
2061  004370  012705  025462                  MOV     #ORDERQ,R5      ;ADDRESS OF ORDER QUEUE IN R5
2062  004374  005725                          TST     (R5)+           ;END OF QUEUE ?
2063  004376  001376                          BNE     .-2             ;BR IF NOT
2064  004400  010045                          MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO QUEUE
2065  004402  000413                          BR      8$              ;COMPRESS THE 'AVAIL' LIST
2066  004404  005360  000064          5$:     DEC     $FAIR(R0)       ;DECREMENT THE 'FAIRNESS' COUNT
2067  004410  001402                          BEQ     6$              ;ENTRY WAITING FOR BUFFER PAST LIMIT
2068  004412  005722                          TST     (R2)+           ;INCREMENT THE 'AVAILABLE TABLE' POINTER
2069  004414  000730                          BR      1$              ;CONTINUE LOOKING
2070  004416  012705  025572          6$:     MOV     #WAIT,R5        ;TABLE ADDRESS
2071  004422  005725                          TST     (R5)+           ;AT END OF TABLE ?
2072  004424  001376                          BNE     .-2             ;BR IF NOT
2073  004426  016245  025550          7$:     MOV     AVAIL(R2),-(R5)  ;MOVE ENTRY FROM AVAILABLE TO WAIT
2074  004432  012701  025550          8$:     MOV     #AVAIL,R1       ;TABLE TO COMPRESS
2075  004436  060201                          ADD     R2,R1           ;FORM TABLE POSITION
2076  004440  004737  006650                  JSR     PC,CMPRES       ;COMPRESS THE INDICATED TABLE
2077  004444  000714                          BR      1$              ;CONTINUE PROCESSING
2078
2079                                  ;GET BUFFER ASSIGNMENTS FOR DRIVES IN THE 'BUFFER WAIT' QUEUE
2080
2081  004446  013700  025572  MAIN3:  MOV     WAIT,R0         ;MOVE THE 'WAIT' ENTRY TO R0
2082  004452  005046                  CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
2083  004454  004737  004720          JSR     PC,GETBUF       ;TRY TO GET A BUFFER
2084  004460  012660  000006          MOV     (SP)+,$BUF(R0)  ;MOVE THE BUFFER ADDR TO THE DPB
2085  004464  001002                  BNE     1$              ;BR IF A BUFFER WAS ASSIGNED
2086  004466  000137  004532          JMP     IDLE            ;NO BUFFER AVAILABLE YET
2087  004472  004737  005304  1$:     JSR     PC,FILBUF       ;FILL THE BUFFER
2088  004476  004737  005434          JSR     PC,GODRIV       ;PUT THE ENTRY IN THE DRIVER
2089  004502  005060  000064          CLR     $FAIR(R0)       ;CLEAR THE 'FAIRNESS' COUNT
2090  004506  012705  025462          MOV     #ORDERQ,R5      ;ADDRESS OF ORDER QUEUE IN R5
2091  004512  005725                  TST     (R5)+           ;AT END OF THE QUEUE
2092  004514  001376                  BNE     .-2             ;BR IF NOT
2093  004516  010045                  MOV     R0,-(R5)        ;PUT BLOCK ADDRESS IN QUEUE
2094  004520  012701  025572          MOV     #WAIT,R1        ;TABLE ADDRESS
2095  004524  004737  006650          JSR     PC,CMPRES       ;COMPRESS THE WAIT TABLE
2096  004530  000656                  BR      MAIN2           ;LOOK FOR MORE ENTRIES
2097
2098                                  ;WAIT FOR AN ORDER TO FINISH
2099
2100  004532  012701  025462  IDLE:  MOV     #ORDERQ,R1      ;ADDRESS OF THE ORDER QUEUE IN R1
2101  004536  012100  1$:     MOV     (R1)+,R0        ;PUT BLOCK ADDRESS INTO R0
2102  004540  001430                  BEQ     3$              ;BR IF END OF QUEUE
2103  004542  005760  000016          TST     $STATUS(R0)     ;SEE IF UNIT FINISHED
2104  004546  001773                  BEQ     1$              ;BR IF UNIT NOT FINISHED
2105  004550  162701  000002          SUB     #2,R1           ;CORRECT THE QUEUE POINTER
2106  004554  010146                  MOV     R1,-(SP)        ;SAVE THE QUEUE ADDRESS
2107  004556  004737  004626          JSR     PC,STATIS       ;ACCUMULATE STATISTICS FOR UNIT IN R0
2108  004562  004737  012374          JSR     PC,PROCES       ;PROCESS END OF ORDER
2109  004566  004737  005056          JSR     PC,RELBUF       ;RESTORE THE BUFFER
2110  004572  004737  011306          JSR     PC,ABNRML       ;SEE IF ANY UNITS HAVE TOO MANY ERRORS
```

```
2111  004576  004737  011334                  JSR     PC,NRML          ;SEE IF ANY UNIT HAS XFERED 3X10↑9 BITS
2112  004602  012601                          MOV     (SP)+,R1         ;RESTORE THE ORDER TABLE INDEX
2113  004604  012705  025550                  MOV     #AVAIL,R5        ;FIND THE END OF THE 'AVAILABLE' TABLE
2114  004610  005725                  2$:     TST     (R5)+            ;END OF THE TABLE ?
2115  004612  001376                          BNE     2$               ;BR IF NOT AT END OF LIST
2116  004614  011145                          MOV     (R1), -(R5)      ;MOV THE BLOCK ADDRESS INTO THE TABLE
2117  004616  004737  006650                  JSR     PC,CMPRES        ;COMPRESS THE ORDER QUEUE
2118  004622  000137  003650          3$:     JMP     MAIN             ;CONTINUE THE LOOP
2119
2120                                          ;ROUTINE TO UPDATE THE GENERAL STATISTICS FOR THE DRIVE IN R0
2121
2122  004626  016037  000076  004716  STATIS: MOV     $RPBA(R0),FACTOR  ;STORE THE FINAL BUFFER ADDRESS
2123  004634  166037  000006  004716          SUB     $BUF(R0),FACTOR ;SUBTRACT THE INITIAL ADDRESS
2124  004642  001413                          BEQ     1$               ;BR IF NO DATA TRANSFER
2125  004644  006237  004716                  ASR     FACTOR           ;CONVERT TO A WORD COUNT
2126  004650  132760  000004  000002          BITB    #BIT02,$COMND(R0)  ;READ OR WRITE ORDER ?
2127  004656  001405                          BEQ     1$               ;BR IF WRITE
2128  004660  063760  004716  000044          ADD     FACTOR,$READ(R0)  ;UPDATE THE READ WORD COUNT
2129  004666  005560  000046                  ADC     $READ+2(R0)
2130  004672  026060  000010  000106  1$:     CMP     $CYL(R0),$SUCA(R0)  ;DID MID-TRANSFER SEEK OCCUR
2131  004700  001405                          BEQ     2$               ;BR IF NOT
2132  004702  062760  000001  000040          ADD     #1,$POSIT(R0)    ;INCREMENT SEEK COUNT
2133  004710  005560  000042                  ADC     $POSIT+2(R0)     ;ADD CARRY TO UPPER WORD
2134  004714  000207                  2$:     RTS     PC
2135
2136  004716  000000                  FACTOR: .WORD   0                ;USED FOR WORDS TRANSFERED
2137
2138                                          ;;*****************************************************************
2139
2140                                          .SBTTL  BUFFER SUPPORT ROUTINES
2141
2142                                          ;;*****************************************************************
2143
2144                                          ;ROUTINE TO GET A BUFFER
2145
2146  004720  010146                  GETBUF: MOV     R1,-(SP)         ;SAVE R1
2147  004722  010246                          MOV     R2,-(SP)         ;SAVE R2
2148  004724  010346                          MOV     R3,-(SP)         ;SAVE R3
2149  004726  013702  025614                  MOV     BUFTBL,R2        ;NUMBER OF SEPARATE BUFFERS
2150  004732  001413                          BEQ     5$               ;BR IF NONE AVAILABLE
2151  004734  012701  025616                  MOV     #BUFTBL+2,R1     ;FIRST ADDRESS OF ALLOCATION TABLE
2152  004740  026061  000020  000002  1$:     CMP     $WRDL(R0),2(R1)  ;SEE IF THERE IS A BLOCK LARGE ENOUGH
2153  004746  101411                          BLOS    3$               ;BRANCH IF IT IS
2154  004750  005302                          DEC     R2               ;DECREMENT TABLE COUNT
2155  004752  001403                          BEQ     5$               ;BR IF THROUGH TABLE
2156  004754  062701  000004                  ADD     #4,R1            ;INCREMENT TABLE POINTER
2157  004760  000767                          BR      1$               ;CONTINUE LOOKING
2158  004762  012603                  5$:     MOV     (SP)+,R3         ;RESTORE R3
2159  004764  012602                          MOV     (SP)+,R2         ;RESTORE R2
2160  004766  012601                          MOV     (SP)+,R1         ;RESTORE R1
2161  004770  000207                          RTS     PC               ;RETURN
2162  004772  011166  000010          3$:     MOV     (R1),10(SP)      ;BUFFER ADDRESS TO STACK
2163  004776  166061  000020  000002          SUB     $WRDL(R0),2(R1)  ;ADJUST BUFFER SIZE
2164  005004  001407                          BEQ     4$               ;BR IF DIFFERENCE IS ZERO
2165  005006  006360  000020                  ASL     $WRDL(R0)        ;CONVERT # WORDS TO BYTES
2166  005012  066011  000020                  ADD     $WRDL(R0),(R1)   ;MAKE NEW STARTING ADDRESS
```

```
2167  005016  006260  000020                    ASR     $WRDL(R0)       ;RETURN # BYTES TO WORDS
2168  005022  000757                            BR      5$              ;RETURN
2169  005024  005337  025614          4$:       DEC     BUFTBL          ;DECREMENT ENTRIES COUNT
2170  005030  001754                            BEQ     5$              ;BR IF ALLOCATION TABLE EMPTY
2171  005032  005302                            DEC     R2              ;DECREMENT TABLE COUNT
2172  005034  001752                            BEQ     5$              ;BR IF ITEM WERE LAST ENTRY
2173  005036  010103                            MOV     R1,R3           ;MOVE TABLE POINTER
2174  005040  062703  000004                    ADD     #4,R3           ;POINT TO NEXT ENTRY
2175  005044  012321                    6$:     MOV     (R3)+,(R1)+     ;MOVE ITEMS
2176  005046  012321                            MOV     (R3)+,(R1)+
2177  005050  005302                            DEC     R2              ;DECREMENT TABLE COUNT
2178  005052  001374                            BNE     6$              ;CONTINUE IF NOT AT END OF TABLE
2179  005054  000742                            BR      5$              ;RETURN
2180
2181
2182                                    ;ROUTINE TO PUT BUFFER BACK IN TABLE
2183
2184  005056  104412                    RELBUF: SAVREG                  ;SAVE R0-R5
2185  005060  012701  025616                    MOV     #BUFTBL+2,R1    ;BEGINNING OF TABLE
2186  005064  013702  025614                    MOV     BUFTBL,R2       ;ENTRY COUNT
2187  005070  001424                            BEQ     2$              ;BR IF EMPTY TABLE
2188  005072  016003  000020                    MOV     $WRDL(R0),R3    ;TRIAL ADDRESS
2189  005076  006303                            ASL     R3              ;CHANGE TO BYTE COUNT
2190  005100  066003  000006                    ADD     $BUF(R0),R3     ;ADDRESS OF HIGHER ADJACENT BLOCK
2191  005104  021103                    1$:     CMP     (R1),R3         ;UPPER ADJACENT BLOCK
2192  005106  001432                            BEQ     4$              ;BR IF YES
2193  005110  062701  000004                    ADD     #4,R1           ;INCREMENT POINTER
2194  005114  005302                            DEC     R2              ;DECREMENT ENTRY COUNT
2195  005116  001372                            BNE     1$              ;CONTINUE SEARCHING
2196  005120  016011  000006                    MOV     $BUF(R0),(R1)   ;PUT THE BUFFER BLOCK INTO THE TABLE
2197  005124  016061  000020  000002            MOV     $WRDL(R0),2(R1)  ;BLOCK SIZE
2198  005132  005237  025614                    INC     BUFTBL          ;INCREMENT ENTRY COUNT
2199  005136  005202                            INC     R2              ;INCREMENT R2 FOR USE LATER
2200  005140  000422                            BR      5$              ;SEE IF A LOWER ADJACENT BLOCK IS IN THE TABLE
2201  005142  016021  000006          2$:       MOV     $BUF(R0),(R1)+  ;BLOCK ADDRESS TO TABLE
2202  005146  016021  000020                    MOV     $WRDL(R0),(R1)+ ;SIZE TO TABLE
2203  005152  005237  025614                    INC     BUFTBL          ;INCREMENT ENTRY COUNT
2204  005156  022737  000002  025614    3$:     CMP     #2,BUFTBL       ;SEE HOW MANY BLOCKS ARE IN TABLE
2205  005164  103001                            BHIS    .+4             ;BR IF NOT MORE THAN 2
2206  005166  000240                            NOP                     ;DEBUGGING AID
```

```
2207  005170  104413                        RESREG                  ;RESTORE R0-R5
2208  005172  000207                        RTS     PC              ;RETURN
2209  005174  016011  000006          4$:   MOV     $BUF(R0),(R1)   ;RELEASED BUFFER IS LOWER ADJACENT
2210  005200  066061  000020  000002        ADD     $WRDL(R0),2(R1) ;INCREMENTED SIZE
2211  005206  010246                  5$:   MOV     R2,-(SP)        ;SAVE R2
2212  005210  013702  025614                MOV     BUFTBL,R2       ;ENTRY COUNT
2213  005214  012705  025616                MOV     #BUFTBL+2,R5    ;BEGINNING OF TABLE
2214  005220  016504  000002          6$:   MOV     2(R5),R4        ;BLOCK SIZE (IN WORDS)
2215  005224  006304                        ASL     R4              ;CHANGE TO BYTE COUNT
2216  005226  061504                        ADD     (R5),R4         ;ADD BLOCK BEGINNING ADDRESS
2217  005230  020411                        CMP     R4,(R1)         ;R1 STILL POINTS TO INSERTED ENTRY
2218  005232  001406                        BEQ     8$              ;LOWER ADJACENT IN TABLE
2219  005234  062705  000004                ADD     #4,R5           ;INCREMENT POINTER
2220  005240  005302                        DEC     R2              ;DECREMENT ENTRY COUNT
2221  005242  001366                        BNE     6$              ;CONTINUE LOOKING
2222  005244  005726                        TST     (SP)+           ;RESTORE STACK POINTER
2223  005246  000743                        BR      3$              ;END
2224  005250  012602                  8$:   MOV     (SP)+,R2        ;RESTORE R2
2225  005252  066165  000002  000002        ADD     2(R1),2(R5)     ;INCREMENT LOWER BLOCK LENGTH
2226  005260  005337  025614                DEC     BUFTBL          ;DECREMENT ENTRY COUNT
2227  005264  010105                        MOV     R1,R5           ;GET READY TO COMPRESS
2228  005266  062705  000004                ADD     #4,R5           ;INCREMENT TO NEXT ENTRY
2229  005272  012521                  9$:   MOV     (R5)+,(R1)+     ;COMPRESS TABLE
2230  005274  012521                        MOV     (R5)+,(R1)+     ;MOVE SIZE FIELD DOWN
2231  005276  005302                        DEC     R2              ;DECREMENT ENTRY COUNT
2232  005300  001374                        BNE     9$              ;BR IF NOT FINISHED
2233  005302  000725                        BR      3$              ;FINISHED
2234
2235
2236                          ;FILL THE ASSIGNED BUFFER (IF WRITE OR WRITE CHECK ORDER)
2237
2238  005304  104412                  FILBUF: SAVREG                  ;SAVE R0-R5
2239  005306  132760  000004  000002        BITB    #BIT02,$COMND(R0)  ;READ OR WRITE ORDER ?
2240  005314  001404                        BEQ     1$              ;BR IF WRITE
2241  005316  122760  000007  000002        CMPB    #WRTCHK,$COMND(R0)  ;WRITE CHECK ?
2242  005324  001041                        BNE     3$              ;BR IF NOT
2243  005326  026027  000006  033644  1$:   CMP     $BUF(R0),#ENDPGM  ;MAKE SURE BUFFER IS ABOVE PROGRAM
2244  005334  103002                        BHIS    .+6             ;BR IF IT IS
2245  005336  000000                        HALT                    ;BUFFER ADDRESS WITHIN PROGRAM, ILLEGAL
2246  005340  000776                        BR      .-2             ;INTERLOCK THE HALT
2247  005342  023760  001306  000020        CMP     BUFRSZ,$WRDL(R0)  ;SEE IF WORD LENGTH DOESN'T EXCEED MAXIMUM
2248  005350  103002                        BHIS    .+6             ;BR IF IT DOESN'T
2249  005352  000000                        HALT                    ;WORD LENGTH FOR OPERATION GREATER THAN AVAIL MEM
2250  005354  000776                        BR      .-2             ;INTERLOCK THE HALT
2251  005356  016001  000006                MOV     $BUF(R0),R1     ;BUFFER ADDRESS
2252  005362  016002  000020                MOV     $WRDL(R0),R2    ;POSITIVE WORD COUNT
2253  005366  005004                        CLR     R4              ;CLEAR R4
2254  005370  116004  000026                MOVB    $PATTC(R0),R4   ;RELATIVE PATTERN ADDRESS
2255  005374  016405  025764                MOV     STNDAT(R4),R5   ;PATTERN ADDRESS
2256  005400  012703  000020                MOV     #16.,R3         ;PATTERN COUNT
2257  005404  012521                  2$:   MOV     (R5)+,(R1)+     ;MOVE THE PATTERN INTO THE BUFFER
2258  005406  005302                        DEC     R2              ;DECREMENT THE WORD COUNT
2259  005410  001407                        BEQ     3$              ;BR IF DONE (WORD COUNT = 0)
2260  005412  005303                        DEC     R3              ;DECREMENT THE PATTERN COUNT
2261  005414  001373                        BNE     2$              ;BR IF MORE PATTERN
2262  005416  012703  000020                MOV     #16.,R3         ;RESTORE PATTERN COUNT
```

```
                                        MOV     STNDAT(R4),R5   ;RESTORE THE ADDRESS
                                        BR      2$              ;CONTINUE DISTRIBUTING THE PATTERN
                                3$:     RESREG                  ;RESTORE R0-R5
                                        RTS     PC              ;RETURN

                                ;START THE ORDER FOR THE DPB IN R0

                                GODRIV: MOV     R0,-(SP)        ;SAVE R0
                                        MOV     R0,1$           ;CURRENT DPB ADDRESS
                                        JSR     R0,RP11         ;DRIVER ENTRANCE
                                1$:     0                       ;DPB ADDR GOES HERE
                                        HALT                    ;DRIVER REJECTED REQUEST
                                        MOV     (SP)+,R0        ;RESTORE R0
                                        ADD     #1,$OPERC(R0)   ;INCREMENT THE OPERATION COUNT
                                        ADC     $OPERC+2(R0)    ;ADD ANY CARRY
                                        CMP     $PREVA+2(R0),$CYL(R0)   ;DID ORDER REQUIRE A CYLINDER CHANGE
                                        BEQ     2$              ;BR IF NOT
                                        ADD     #1,$POSIT(R0)   ;INCREMENT SEEK COUNT
                                        ADC     $POSIT+2(R0)    ;ADD ANY CARRY
                                2$:     RTS     PC

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

                                .SBTTL  GENERATE VARIABLES FOR THE NEXT OPERATION

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

                                ;GENERATE VARIABLES FOR THE OPERATION

                                GETPAR: MOV     R5,-(SP)        ;SAVE R5
                                        JSR     PC,PREVST       ;SAVE CURRENT SECTOR & TRACK ADDRS
                                        MOV     $SUCA(R0),$PREVA+2(R0)  ;SAVE CURRENT CYLINDER
                                        BIT     #SW05,@SWR      ;SWITCH 5 SET ?
                                        BEQ     GTPAR1          ;BR IF NOT SET
                                        CMPB    #WRTCHK,$COMND(R0)  ;PRESENT ORDER A WRITE CHECK ?
                                        BNE     1$              ;BR IF NOT
                                        TST     AUTOCK          ;AUTO WRITE CHECKS ?
                                        BEQ     1$              ;BR IF NOT
                                        MOVB    $PREVO(R0),$COMND(R0)   ;RESTORE PREVIOUS OPERATION
                                1$:     JMP     PARXIT          ;EXIT
                                GTPAR1: MOVB    $COMND(R0),$PREVO(R0)   ;SAVE CURRENT OPERATION CODE
                                        JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
                                        BIT     #SW0,@SWR       ;SEE IF SW0 SET
                                        BNE     2$              ;BR IF SET - READ ONLY
                                1$:     MOV     #10,R5          ;READ/WRITE SELECTION DIVISOR
                                        JSR     PC,GETREM       ;GET SELECTION VALUE
                                        CMP     R5,RATIO        ;DETERMINE IF READ OR WRITE
                                        BGT     2$              ;BR IF READ
                                        JSR     PC,RANWRT       ;SELECT A WRITE ORDER
                                        CMPB    #WRTCHK,$COMND(R0)  ;WAS WRITE CHECK SELECTED ?
                                        BNE     GETSEC          ;NO, CONTINUE WITH THE SELECTION
                                        JMP     PARXIT          ;BYPASS REST OF VARIABLE SELECTION
                                2$:     MOV     $HINUM,R5       ;SELECT READ OPERATION CODE
                                        BIC     #^C1,R5         ;MASK OUT ALL BUT BIT 0
                                        ADD     #3,R5           ;TABLE OFFSET FOR READ CODE
                                3$:     MOVB    COMTBL(R5),$COMND(R0)   ;ORDER SELECTION CODE TO CONTROL BLOCK
```

```
2319
2320                                    ;GENERATE A RANDOM SECTOR ADDRESS BETWEEN VALUES 'FS' & 'LS'
2321
2322  005670  013705  001334    GETSEC: MOV     LS,R5   ;GET MAXIMUM SECTOR ADDRESS
2323  005674  163705  001336            SUB     FS,R5   ;SUBTRACT MINIMUM SECTOR ADDRESS
2324  005700  001403                    BEQ     1$              ;BR IF MIN & MAX THE SAME
2325  005702  005205                    INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
2326  005704  004737  011600            JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
2327  005710  063705  001336    1$:     ADD     FS,R5   ;CONVERT TO ADDRESS
2328  005714  110560  000012            MOVB    R5,$SEC(R0)     ;STORE SECTOR ADDRESS IN DPB
2329
2330                                    ;GENERATE A RANDOM TRACK ADDRESS BETWEEN VALUES 'FT' & 'LT'
2331
2332  005720  013705  001330    GETRK:  MOV     LT,R5   ;GET MAXIMUM TRACK ADDRESS
2333  005724  163705  001332            SUB     FT,R5   ;SUBTRACT MINIMUM TRACK ADDRESS
2334  005730  001403                    BEQ     1$              ;BR IF MIN & MAX THE SAME
2335  005732  005205                    INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
2336  005734  004737  011600            JSR     PC,GETREM       ;GET THE RANDOM TRACK AUGMENT
2337  005740  063705  001332    1$:     ADD     FT,R5   ;CONVERT AUGMENT TO AN ADDRESS
2338  005744  110560  000013            MOVB    R5,$TRK(R0)     ;STORE TRACK ADDRESS IN DPB
2339
2340                                    ;GENERATE A RANDOM CYLINDER ADDRESS BETWEEN VALUES 'FC' & 'LC'
2341
2342  005750  013705  001324    GETCYL: MOV     LC,R5   ;GET MAXIMUM CYLINDER ADDRESS
2343  005754  163705  001326            SUB     FC,R5   ;SUBTRACT MINIMUM CYLINDER ADDRESS
2344  005760  001403                    BEQ     1$              ;BR IF MIN & MAX THE SAME
2345  005762  005205                    INC     R5              ;INCREMENT DIFFERENCE TO USE AS DIVISOR
2346  005764  004737  011600            JSR     PC,GETREM       ;GET THE RANDOM AUGMENT
2347  005770  063705  001326    1$:     ADD     FC,R5   ;CONVERT AUGMENT TO AN ADDRESS
2348  005774  010560  000010            MOV     R5,$CYL(R0)     ;;STORE CYLINDER ADDRESS IN DPB
2349
2350                                    ;GENERATE A RANDOM BUFFER LENGTH BETWEEN 4 & THE VALUE IN 'BUFRSZ'
2351
2352  006000  013705  001306    GETSIZ: MOV     BUFRSZ,R5       ;GET BUFFER SIZE
2353  006004  005205                    INC     R5              ;INCREMENT THE MAXIMUM SIZE
2354  006006  004737  011600            JSR     PC,GETREM       ;DIVIDE BY MAX VALUE
2355  006012  005705                    TST     R5              ;IS THE REMAINDER 0 ?
2356  006014  001413                    BEQ     1$              ;ZERO, TRY AGAIN
2357  006016  010546                    MOV     R5,-(SP)        ;NEW WORD LENGTH ON STACK FOR CHECK
2358  006020  005046                    CLR     -(SP)           ;MAKE UPPER DIVIDEND ZERO
2359  006022  012746  000400            MOV     #256.,-(SP)     ;SECTOR SIZE IS THE DIVISOR
2360  006026  004737  020206            JSR     PC,$DIV         ;DIVIDE BUFFER SIZE BY SECTOR SIZE
2361  006032  021627  000004            CMP     (SP),#4         ;SEE IF REMAINDER LESS THAN 4
2362  006036  103005                    BHIS    2$              ;BR IF NOT
2363  006040  062706  000004            ADD     #4,SP           ;CORRECT THE STACK POINTER
2364  006044  004737  020522    1$:     JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
2365  006050  000753                    BR      GETSIZ          ;TRY AGAIN
2366  006052  132760  000004  000002  2$:     BITB    #4,$COMND(R0)   ;READ OR WRITE ?
2367  006060  001002                    BNE     3$              ;BR IF READ
2368  006062  042705  000001            BIC     #1,R5           ;MAKE THE SIZE EVEN
2369  006066  010560  000020    3$:     MOV     R5,$WRDL(R0)    ;WORD LENGTH TO CONTROL BLOCK
2370  006072  122760  000013  000002            CMPB    #WRITE,$COMND(R0)  ;WRITE WITHOUT IMPLIED SEEK ?
2371  006100  001404                    BEQ     4$              ;BR IF IT IS
2372  006102  122760  000017  000002            CMPB    #READ,$COMND(R0)   ;READ WITHOUT IMPLIED SEEK ?
2373  006110  001027                    BNE     6$              ;BR IF NOT
2374  006112  005016            4$:     CLR     (SP)            ;CLEAR THE STACK
```

```
2375  006114  116016  000012          MOVB    $SEC(R0),(SP)   ;MOVE THE SECTOR ADDRESS
2376  006120  066616  000002          ADD     2(SP),(SP)      ;ADD THE NUMBER OF SECTORS IN THE BUFFER
2377  006124  022716  000010          CMP     #8.,(SP)        ;DO THEY EXCEED THE TRACK
2378  006130  103017                  BHIS    6$              ;BR IF NOT
2379  006132  132760  000010  000002  BITB    #10,$COMND(R0)  ;SEE IF AN 'IMPLIED SEEK' COMMAND
2380  006140  001413                  BEQ     6$              ;BR IF NOT
2381  006142  132760  000004  000002  BITB    #4,$COMND(R0)   ;READ WITHOUT IMPLIED SEEK ?
2382  006150  001004                  BNE     5$              ;BR IF READ
2383  006152  112760  000003  000002  MOVB    #WRTSEK,$COMND(R0)  ;LOAD COMMAND
2384  006160  000403                  BR      6$              ;CONTINUE
2385  006162  112760  000005  000002  5$:     MOVB    #RDSEK,$COMND(R0)   ;LOAD COMMAND
2386  006170  062706  000004          6$:     ADD     #4,SP           ;CORRECT THE STACK POINTER
2387  006174  016060  000020  000004          MOV     $WRDL(R0),$WRDM(R0) ;SETUP WORD COUNT
2388  006202  005460  000004          NEG     $WRDM(R0)       ;MAKE IT NEGATIVE
2389
2390                          ;GET THE PATTERN CODE & EXIT FROM PARAMETER SELECTION
2391
2392  006206  004737  006304          PATCOD: JSR     PC,GETPAT       ;GET PATTERN CODE
2393  006212  110560  000026                  MOVB    R5,$PATTC(R0)   ;MOVE PATTERN CODE TO CONTROL BLOCK
2394  006216  013760  001254  000064  PARXIT: MOV     FAIRNS,$FAIR(R0)  ;LOAD 'FAIRNESS' COUNT
2395  006224  012605                  MOV     (SP)+,R5        ;RESTORE R5
2396  006226  000207                  RTS     PC              ;RETURN
2397
2398                          ;ROUTINE TO SELECT A WRITE (OR WRITE CHECK) OPERATION
2399
2400  006230  012705  000003          RANWRT: MOV     #3,R5           ;WRITE OPERATION SELECTION DIVISOR
2401  006234  004737  011600                  JSR     PC,GETREM       ;GET SELECTION CODE
2402  006240  005737  001322                  TST     AUTOCK          ;ARE WRITE CHECK ORDERS TO BE SELECTED
2403                                          ;RANDOMLY ?
2404  006244  001403                  BEQ     1$              ;BR IF THEY ARE
2405  006246  042705  000002          BIC     #2,R5           ;SET CODE TO EXCLUDE WRITE CHECK ORDERS
2406  006252  000410                  BR      2$              ;FINISH
2407  006254  020527  000002          1$:     CMP     R5,#2           ;WRITE CHECK SELECTED ?
2408  006260  001005                  BNE     2$              ;BR IF NOT
2409  006262  132760  000004  000024          BITB    #BIT02,$PREVO(R0)   ;WAS PREVIOUS ORDER A WRITE ?
2410  006270  001401                  BEQ     2$              ;BR IF IT WAS
2411  006272  005005                  CLR     R5              ;CHANGE WRITE CHECK CODE TO WRITE WITH
2412                                          ;IMPLIED SEEK
2413  006274  116560  025756  000002  2$:     MOVB    COMTBL(R5),$COMND(R0)   ;CODE TO DRIVE BLOCK
2414  006302  000207                  RTS     PC              ;RETURN
2415
2416                          ;ROUTINE TO SELECT A PATTERN
2417
2418  006304  012705  000020          GETPAT: MOV     #16.,R5         ;SELECT PATTERN
2419  006310  004737  011600                  JSR     PC,GETREM       ;GET CODE
2420  006314  005705                  TST     R5              ;WAS PATTERN ZERO SELECTED ?
2421  006316  001003                  BNE     1$              ;BR IF NOT ZERO
2422  006320  004737  020522                  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
2423  006324  000767                  BR      GETPAT          ;TRY AGAIN
2424  006326  006305                  1$:     ASL     R5              ;MAKE CODE INTO TABLE INDEX
2425  006330  000207                  RTS     PC              ;RETURN
2426
2427                          ;ROUTINE TO SETUP PARAMETERS FOR A SEQUENTIAL READ OR WRITE OF THE DISK
2428
2429  006332  004737  020522          WRTPK:  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
2430  006336  005760  000036                  TST     $OPERC+2(R0)    ;SEE IF FIRST OPERATION
```

```
2431  006342  001007                          BNE     WRTPK1          ;BR IF UPPER WORD OF COUNTER NOT ZERO
2432  006344  005760  000034                  TST     $OPERC(R0)      ;LOWER WORD ZERO ?
2433  006350  001004                          BNE     WRTPK1          ;BR IF NOT 1ST OPERATION
2434  006352  105760  000022                  TSTB    $PACK(R0)       ;SEE WHICH - 'R' OR 'W'
2435  006356  100504                          BMI     WRTPK3          ;BR IF 'W'
2436  006360  000477                          BR      WRTPK2          ;'R' OPERATION
2437  006362  032777  000040  172550  WRTPK1: BIT     #SWO5,@SWR      ;LOOP ON LAST ADDRESS ?
2438  006370  001106                          BNE     WRTPK4          ;BR IF LOOP
2439  006372  032760  000002  000070          BIT     #EOP,$RPER(R0)  ;DID LAST ORDER OVERFLOW ?
2440  006400  001032                          BNE     1$              ;BR IF IT DID
2441  006402  116060  000002  000024          MOVB    $COMND(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
2442  006410  004737  006616                  JSR     PC,PREVST       ;SAVE CURRENT SECTOR & TRACK
2443  006414  016060  000106  000032          MOV     $SUCA(R0),$PREVA+2(R0)  ;CURRENT CYLINDER
2444  006422  016060  000102  000012          MOV     $RPDA(R0),$SEC(R0)  ;NEW SECTOR & TRACK ADDRESS
2445  006430  042760  000360  000012          BIC     #360,$SEC(R0)   ;CLEAR THE 'SOT' BITS
2446  006436  016060  000100  000010          MOV     $RPCA(R0),$CYL(R0)  ;NEW CYLINDER ADDRESS
2447  006444  026037  000010  001324          CMP     $CYL(R0),LC     ;SEE IF END CYLINDER
2448  006452  103427                          BLO     2$              ;BR IF NOT
2449  006454  101004                          BHI     1$              ;BR IF PAST END CYLINDER
2450  006456  126037  000013  001330          CMPB    $TRK(R0),LT     ;END TRACK ?
2451  006464  101422                          BLOS    2$              ;BR IF NOT EXCEEDED
2452  006466  113760  001332  000013  1$:     MOVB    FT,$TRK(R0)     ;RESET TRACK ADDRESS
2453  006474  113760  001336  000012          MOVB    FS,$SEC(R0)  ;RESET SECTOR ADDRESS
2454  006502  013760  001326  000010          MOV     FC,$CYL(R0)  ;RESET CYLINDER ADDRESS
2455  006510  004737  011410                  JSR     PC,NRML2        ;DROP THE UNIT (NORMAL TERMINATION)
2456  006514  032777  000020  172416          BIT     #SW04,@SWR      ;IS SWITCH 4 SET ?
2457  006522  001003                          BNE     2$              ;BR IF SET
2458  006524  005726                          TST     (SP)+           ;INCREMENT THE STACK POINTER
2459  006526  000137  003650                  JMP     MAIN            ;RETURN DIRECTLY TO 'MAIN'
2460  006532  013760  001306  000020  2$:     MOV     BUFRSZ,$WRDL(R0)  ;BUFFER SIZE IS MAXIMUM
2461  006540  013760  001306  000004          MOV     BUFRSZ,$WRDM(R0)  ;WORD COUNT
2462  006546  005460  000004                  NEG     $WRDM(R0)       ;CHANGE WORD COUNT TO 2'S COMPLEMENT
2463  006552  105760  000022                  TSTB    $PACK(R0)       ;READ OR WRITE ?
2464  006556  100404                          BMI     WRTPK3          ;BR IF WRITE
2465  006560  112760  000005  000002  WRTPK2: MOVB    #ROSEK,$COMND(R0)  ;OP CODE
2466  006566  000407                          BR      WRTPK4          ;EXIT
2467  006570  112760  000003  000002  WRTPK3: MOVB    #WRTSEK,$COMND(R0)  ;OP CODE
2468  006576  004737  006304                  JSR     PC,GETPAT       ;GET PATTERN CODE
2469  006602  110560  000026                  MOVB    R5,$PATTC(R0)   ;PATTERN CODE
2470  006606  013760  001254  000064  WRTPK4: MOV     FAIRNS,$FAIR(R0)  ;LOAD 'FAIRNES' COUNT
2471  006614  000207                          RTS     PC              ;RETURN
2472
2473                                  ;DECREMENT AND SAVE CURRENT SECTOR & TRACK ADDRESS
2474
2475  006616  162706  000004          PREVST: SUB     #4,SP           ;MAKE ROOM ON THE STACK
2476  006622  004737  006666                  JSR     PC,READDR       ;DECREMENT SECTOR-TRACK ADDRESS
2477  006626  112660  000031                  MOVB    (SP)+,$PREVA+1(R0)  ;SAVE CURRENT TRACK
2478  006632  112660  000030                  MOVB    (SP)+,$PREVA(R0)  ;SAVE CURRENT SECTOR
2479  006636  000207                          RTS     PC              ;RETURN
2480
2481                                  ;;*****************************************************************
2482
2483                                  .SBTTL  GENERAL SUPPORT SUBROUTINES
2484
2485                                  ;;*****************************************************************
2486
```

```
2487                                    ;DUMMY HANDLER IF 'CONTROL C' ENTERED DURING NORMAL OPERATION
2488
2489  006640  012746  000003            DORTI:  MOV     #3,-(SP)        ;PUT THE 'CONTROL C' BACK ON THE STACK
2490  006644  000137  017100                    JMP     $TKSRV+52       ;RETURN TO THE ENTRY ROUTINE
2491
2492                                    ;ROUTINE TO COMPRESS THE TABLE WHOSE ADDRESS IS IN R1
2493
2494  006650  016111  000002            CMPRES: MOV     2(R1),(R1)      ;COMPRESS THE TABLE IN R1
2495  006654  001403                            BEQ     1$              ;BR WHEN ZERO FOUND
2496  006656  062701  000002                    ADD     #2,R1           ;INCREMENT R1
2497  006662  000772                            BR      CMPRES          ;CONTINUE COMPRESSING TABLE
2498  006664  000207                    1$:     RTS     PC              ;RETURN
2499
2500                                    ;DECREMENT THE SECTOR-TRACK ADDRESS
2501
2502  006666  005066  000004            READDR: CLR     4(SP)           ;CLEAR STACK FOR SECTOR
2503  006672  116066  000102  000004            MOVB    $RPDA(R0),4(SP) ;INCREMENTED SECTOR ON STACK
2504  006700  042766  177760  000004            BIC     #^C17,4(SP)     ;CLEAR THE 'SOT' BITS
2505  006706  105366  000004                    DECB    4(SP)           ;DECREMENT THE SECTOR ADDRESS
2506  006712  100017                            BPL     1$              ;BR IF SECTOR GREATER THAN 0
2507  006714  012766  000011  000004            MOV     #9.,4(SP)       ;JAM SECTOR ADDRESS TO 9
2508  006722  005066  000002                    CLR     2(SP)           ;CLEAR STACK FOR TRACK
2509  006726  116066  000103  000002            MOVB    $RPDA+1(R0),2(SP) ;TRACK ADDRESS
2510  006734  105366  000002                    DECB    2(SP)           ;DECREMENT TRACK ADDRESS
2511  006740  100007                            BPL     2$              ;BR IF IT DIDN'T GO NEG
2512  006742  012766  000023  000002            MOV     #19.,2(SP)      ;RESET TRACK TO 19(10)
2513  006750  000403                            BR      2$
2514  006752  116066  000103  000002    1$:     MOVB    $RPDA+1(R0),2(SP) ;TRACK ADDRESS
2515  006760  000207                    2$:     RTS     PC              ;RETURN
2516
2517                                    ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
2518
2519  006762  012737  177777  001236    CKCLK:  MOV     #-1,CLKFLG      ;CLEAR CLOCK AVAILABILITY FLAG
2520  006770  012737  007044  000004            MOV     #CKCLK1,@#ERRVEC ;SET UP VECTOR FOR CLOCK CHECK
2521  006776  005037  000006                    CLR     @#ERRVEC+2      ;NEW PSW
2522  007002  005777  172216                    TST     @$LKCSR         ;CHECK FOR KW11-P
2523  007006  005037  001236                    CLR     CLKFLG          ;SET CLOCK AVAILABILITY FLAG
2524  007012  013701  001230                    MOV     $LPVEC,R1       ;KW11-P VECTOR ADDRESS
2525  007016  012721  007660                    MOV     #CLOCK,(R1)+    ;SET UP KW11-P VECTOR
2526  007022  012711  000300                    MOV     #300,(R1)       ;PSW - PRI 6
2527  007026  012777  177777  172172            MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER WITH 1'S
2528  007034  012777  000135  172162            MOV     #135,@$LKCSR    ;SET CLOCK - CNT UP, 16MS, CONT INT
2529  007042  000425                            BR      CKCLK3
2530  007044  062706  000004    CKCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
2531  007050  012737  007112  000004            MOV     #CKCLK2,@#ERRVEC ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
2532  007056  005777  172150                    TST     @$LKS           ;LOOK FOR KW11-L
2533  007062  005037  001236                    CLR     CLKFLG          ;SET CLOCK FLAG
2534  007066  013701  001234                    MOV     $LLVEC,R1       ;KW11-L VECTOR ADDRESS
2535  007072  012721  007660                    MOV     #CLOCK,(R1)+    ;SET UP KW11-L VECTOR
2536  007076  012711  000300                    MOV     #300,(R1)       ;PSW - PRI 6
2537  007102  012777  000100  172122            MOV     #100,@$LKS      ;SET KW11-L INTERRUPT
2538  007110  000402                            BR      CKCLK3
2539  007112  062706  000004    CKCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
2540  007116  012737  000006  000004    CKCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
2541  007124  000207                            RTS     PC
2542
```

```
                                        ;ROUTINE TO DISPLAY STATISTICS FOR ALL DRIVES ASSIGNED
                                        
007126  104412                  STATPR: SAVREG                  ;SAVE R0-R5
007130  005004                          CLR     R4              ;R4 CONTAINS THE UNIT POINTER
007132  012703  000010                  MOV     #8.,R3          ;R3 CONTAINS THE UNIT COUNTER
007136  012700  024274                  MOV     #BLKD,R0        ;ADDR OF FIRST COUNTER
007142  136437  021304  025460  1$:     BITB    ATABIT(R4),ASNLST  ;SEE IF DRIVE ASSIGNED
007150  001006                          BNE     2$              ;BR IF IT IS
007152  005204                          INC     R4              ;INCREMENT THE UNIT POINTER
007154  062700  000112                  ADD     #$SILO+2,R0     ;INCREMENT THE BLOCK ADDRESS
007160  005303                          DEC     R3              ;FINISHED ?
007162  001367                          BNE     1$              ;BR IF NOT
007164  000423                          BR      6$              ;YES, EXIT
007166  004737  007270          2$:     JSR     PC,SHDTYP       ;TYPE THE HEADING
007172  136437  021304  025460  3$:     BITB    ATABIT(R4),ASNLST  ;IS THE UNIT ASSIGNED
007200  001003                          BNE     4$              ;BR IF ASSIGNED
007202  062700  000112                  ADD     #$SILO+2,R0     ;ADD TABLE SIZE TO ADDRESS
007206  000407                          BR      5$
007210  010002                  4$:     MOV     R0,R2           ;PUT BLOCK ADDRESS IN R2
007212  062702  000034                  ADD     #$OPERC,R2      ;FIRST COUNTER TO DISPLAY
007216  004737  007306                  JSR     PC,SDETAL       ;TYPE THE STATISTICS
007222  062700  000112                  ADD     #$SILO+2,R0     ;INCREMENT BLOCK ADDRESS
007226  005204                  5$:     INC     R4              ;INCREMENT UNIT NUMBER
007230  005303                          DEC     R3              ;DECREMENT UNIT COUNTER
007232  001357                          BNE     3$              ;BR IF NOT FINISHED
007234  104413                  6$:     RESREG                  ;RESTORE R0-R5
007236  000207                          RTS     PC
                                        
                                        ;ROUTINE TO TYPE STATISTICS FOR THE DRIVE IN R0
                                        
007240  104412                  TYPEST: SAVREG                  ;SAVE R0-R5
007242  004737  007270                  JSR     PC,SHDTYP       ;TYPE THE HEADING
007246  005004                          CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
007250  111004                          MOVB    (R0),R4         ;DRIVE NUMBER
007252  010002                          MOV     R0,R2           ;PUT BLOCK ADDRESS IN R2
007254  062702  000034                  ADD     #$OPERC,R2      ;ADDRESS OF '$OPERC' IN PRESENT CLOCK
007260  004737  007306                  JSR     PC,SDETAL       ;TYPE THE STATISTICS
007264  104413                          RESREG                  ;RESTORE R0-R5
007266  000207                          RTS     PC              ;RETURN
                                        
                                        ;TYPE THE HEADER FOR THE DRIVE PERFORMANCE SUMMARY TYPEOUT
                                        
007270  104401  001213          SHDTYP: TYPE    ,$CRLF          ;CR-LF
007274  004737  007634                  JSR     PC,$TIME        ;TYPE THE TIME OF DAY
007300  104401  032646                  TYPE    ,$TATHD         ;HEADER
007304  000207                          RTS     PC              ;RETURN
                                        
                                        ;TYPE THE DRIVE PERFORMANCE DATA LINE
                                        
007306  010446                  SDETAL: MOV     R4,-(SP)        ;DRIVE NUMBER
007310  104403                          TYPOS                   ;TYPE IT
007312     001     000                  .BYTE   1,0
007314  104401  032477                  TYPE    ,$SPACE3        ;SPACES
007320  016046  000062                  MOV     $PASSC(R0),-(SP)  ;PUT THE PASS COUNT ON THE STACK
007324  004737  015574                  JSR     PC,$S32D        ;CONVERT IT
007330  004737  011642                  JSR     PC,$RPZRS       ;TYPE IT
```

```
                                                TYPE    $SPACE2         ;SPACES
                                                MOV     R2,-(SP)        ;PUT $OPERC ON THE STACK
                                                JSR     PC,$DB2D        ;CONVERT IT
                                                JSR     PC,$RPZR8       ;TYPE $OPERC
                                                TYPE    $SPACE2         ;SPACES
                                                ADD     #4,R2           ;INCREMENT R2
                                                MOV     R2,-(SP)        ;PUT $POSIT ON THE STACK
                                                JSR     PC,$DB2D        ;CONVERT IT
                                                JSR     PC,$RPZR8       ;TYPE $POSIT
                                                TYPE    $SPACE2         ;SPACES
                                                ADD     #4,R2           ;INCREMENT R2
                                                MOV     R2,-(SP)        ;PUT $READ ON THE STACK
                                                JSR     PC,$DB2D        ;CONVERT $READ
                                                JSR     PC,$RPZR0       ;TYPE IT
                                                TYPE    $SPACE1         ;1 SPACE
                                                ADD     #4,R2           ;INCREMENT R2
                                        1$:     MOV     R1,-(SP)        ;SAVE R1
                                                MOV     #4,R1           ;USE R1 AS A COUNTER
                                                ADD     #2,R2           ;INCREMENT R2 AGAIN
                                        2$:     MOV     (R2)+,-(SP)     ;PUT '$SOFT'  '$HARD', '$SKI', OR '$MISPO'
                                                                        ;ON THE STACK
                                                JSR     PC,$SB2D        ;CONVERT IT
                                                JSR     PC,$RPZR4       ;TYPEOUT
                                                TYPE    $SPACE1         ;1 SPACE
                                                DEC     R1              ;DONE ?
                                                BNE     2$              ;BR IF NOT
                                                MOV     (SP)+,R1        ;RESTORE R1
                                                MOV     $TOTAL(R0),-(SP)  ;CALCULATE NUMBER OF OTHER ERRORS
                                                SUB     $SOFT(R0),(SP)  ;SUBTRACT $SOFT FROM $TOTAL
                                                SUB     $HARD(R0),(SP)  ;SUBTRACT $HARD FROM $TOTAL
                                                SUB     $SKI(R0),(SP)   ;SUBTRACT $SKI FROM $TOTAL
                                                SUB     $MISPO(R0),(SP) ;SUBTRACT $MISPO FROM $TOTAL
                                                JSR     PC,$SB2D        ;CONVERT 'OTHER' COUNT
                                                JSR     PC,$RPZR4       ;TYPE IT
                                                TYPE    $CRLF
                                                RTS     PC

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

007526  026027  000052  023417  INCSOF: CMP     $SOFT(R0),#9999.  ;IS $SOFT ALREADY AT MAXIMUM ?
007534  103002                          BHIS    1$              ;BR IF IT IS
007536  005260  000052                  INC     $SOFT(R0)       ;INCREMENT $SOFT
007542  000207                  1$:     RTS     PC              ;RETURN

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

007544  026027  000054  023417  INCHRD: CMP     $HARD(R0),#9999.  ;IS $HARD ALREADY AT MAXIMUM ?
007552  103002                          BHIS    1$              ;BR IF IT IS
007554  005260  000054                  INC     $HARD(R0)       ;INCREMENT $HARD
007560  000207                  1$:     RTS     PC              ;RETURN

                                        ;ROUTINE TO INCREMENT $SKI
```

```
                                        ;
                                        ;NOTE:  $SKI WILL NOT BE INCREMENTED BEYOND 9999 (10)
                                        ;
007562  026027  000056  023417 INCSKI: CMP     $SKI(R0),#9999.  ;IS $SKI ALREADY AT MAXIMUM ?
007570  103002                         BHIS    1$               ;BR IF IT IS
007572  005260  000056                 INC     $SKI(R0)         ;INCREMENT $SKI
007576  000207                 1$:     RTS     PC               ;RETURN
                                        ;
                                        ;ROUTINE TO INCREMENT $MISPO
                                        ;
                                        ;NOTE:  $MISPO WILL NOT BE INCREMENTED BEYOND 9999 (10)
                                        ;
007600  026027  000060  023417 INCMIS: CMP     $MISPO(R0),#9999.  ;IS $MISPO ALREADY AT MAXIMUM ?
007606  103002                         BHIS    1$               ;BR IF IT IS
007610  005260  000060                 INC     $MISPO(R0)       ;INCREMENT $MISPO
007614  000207                 1$:     RTS     PC               ;RETURN
                                        ;
                                        ;ROUTINE TO INCREMENT $TOTAL
                                        ;
                                        ;NOTE:  $TOTAL WILL NOT BE INCREMENTED BEYOND 9999 (10)
                                        ;
007616  026027  000050  023417 INCTOT: CMP     $TOTAL(R0),#9999.  ;IS $TOTAL ALREADY AT MAXIMUM ?
007624  103002                         BHIS    1$               ;BR IF IT IS
007626  005260  000050                 INC     $TOTAL(R0)       ;INCREMENT $TOTAL
007632  000207                 1$:     RTS     PC               ;RETURN
                                        ;
                                        ;ROUTINE TO TYPE THE TIME
                                        ;
007634  005737  001236         $TIME:  TST     CLKFLG           ;CLOCK ON THE SYSTEM ?
007640  001006                         BNE     1$               ;BR IF NOT
007642  104401  010130                 TYPE    ,HOUR            ;TYPE THE HOURS
007646  104401  010134                 TYPE    ,MINUTE          ;TYPE THE MINUTES
007652  104401  010140                 TYPE    ,SECOND          ;TYPE THE SECONDS
007656  000207                 1$:     RTS     PC
                                        ;
                                        ;CLOCK HANDLER ROUTINE
                                        ;
007660  010046                 CLOCK:  MOV     R0,-(SP)         ;SAVE R0
007662  005237  010144                 INC     SIXTEE           ;INCREMENT THE 1/60 SECOND COUNTER
007666  001076                         BNE     1$               ;BR IF A SECOND NOT COUNTED
007670  013737  001240  010144         MOV     #Z,SIXTEE        ;RESTORE THE VALUE
007676  005437  010144                 NEG     SIXTEE           ;MAKE IT INTO 2'S COMP
007702  105237  010141                 INCB    SECOND+1         ;INCREMENT THE SECONDS
007706  122737  000072  010141         CMPB    #72,SECOND+1     ;SEE IF COUNTER AT MAX
007714  001063                         BNE     1$               ;BR IF NOT
007716  112737  000060  010141         MOVB    #60,SECOND+1     ;RESTORE VALUE
007724  105237  010140                 INCB    SECOND           ;INCREMENT UPPER
007730  122737  000066  010140         CMPB    #66,SECOND       ;UPPER AT MAX
007736  001052                         BNE     1$               ;BR IF NOT
007740  112737  000060  010140         MOVB    #60,SECOND       ;RESET UPPER
007746  105237  001313                 INCB    INTRVL+1         ;INCREMENT STATISTICS TYPEOUT INTERVAL
007752  105237  010135                 INCB    MINUTE+1         ;INCREMENT MINUTE
007756  122737  000072  010135         CMPB    #72,MINUTE+1     ;AT MAX ?
007764  001037                         BNE     1$               ;BR IF NOT
007766  112737  000060  010135         MOVB    #60,MINUTE+1     ;RESET LOWER
```

```
2711  007774  105237  010134                    INCB    MINUTE          ;INCREMENT UPPER HALF
2712  010000  122737  000066  010134            CMPB    #66,MINUTE      ;AT MAX ?
2713  010006  001026                            BNE     1$              ;BR IF NOT
2714  010010  112737  000060  010134            MOVB    #60,MINUTE      ;RESET UPPER
2715  010016  105237  010131                    INCB    HOUR+1          ;INCREMENT HOURS
2716  010022  022737  035071  010130            CMP     #35071,HOUR     ;AT MAX (99) ?
2717  010030  001412                            BEQ     2$              ;BR IF IT IS
2718  010032  122737  000072  010131            CMPB    #72,HOUR+1      ;AT MAX ?
2719  010040  001011                            BNE     1$              ;BR IF NOT
2720  010042  112737  000060  010131            MOVB    #60,HOUR+1      ;RESET VALUE
2721  010050  105237  010130                    INCB    HOUR            ;INCREMENT UPPER
2722  010054  000403                            BR      1$
2723  010056  012737  030060  010130    2$:     MOV     #30060,HOUR     ;RESET HOUR FIELD
2724  010064  012746  000021            1$:     MOV     #17,-(SP)       ;17 MS ON THE STACK
2725  010070  004737  023360                    JSR     PC,RPTMR        ;DRIVER TIMER ROUTINE
2726  010074  105737  001312                    TSTB    INTRVL          ;DISPLAY THE PERFORMANCE SUMMARY ?
2727  010100  001411                            BEQ     3$              ;BR IF NOT
2728  010102  123737  001312  001313            CMPB    INTRVL,INTRVL+1 ;DISPLAY INTERVAL FINISHED ?
2729  010110  001005                            BNE     3$              ;BR IF NOT
2730  010112  105037  001313                    CLRB    INTRVL+1        ;RESTORE THE INTERVAL
2731  010116  012737  177777  001250            MOV     #-1,STATIN      ;SET ERROR RATE DATA DISPLAY FLAG
2732  010124  012600                    3$:     MOV     (SP)+,R0        ;RESTORE R0
2733  010126  000002                            RTI
2734
2735  010130      060      060           HOUR:   .BYTE   60,60           ;HOURS
2736  010132      072      000                   .BYTE   72,0            ;':' & TERMINATOR
2737  010134      060      060           MINUTE: .BYTE   60,60           ;MINUTES
2738  010136      072      000                   .BYTE   72,0            ;':' & TERMINATOR
2739  010140      060      060           SECOND: .BYTE   60,60           ;SECONDS
2740  010142      040      000                   .BYTE   40,0            ;SPACE, TERMINATOR
2741                                             .EVEN
2742
2743  010144  000000                    SIXTEE: .WORD   0               ;1/60TH OR 1/50TH OF A SECOND COUNTER
2744
2745                                     ;COMMAND DECODE ROUTINE
2746
2747  010146  104412                    KSR:    SAVREG                  ;SAVE R0-R5
2748  010150  012746  000200                    MOV     #PR4,-(SP)      ;RETURN AT PRIORITY 4
2749  010154  012746  010230                    MOV     #KSR1,-(SP)     ;DUMMY INTERRUPT RETURN
2750  010160  017746  170762                    MOV     @#TKB,-(SP)     ;GET THE CHARACTER FROM THE KEYBOARD
2751  010164  042716  177600                    BIC     #^C177,(SP)     ;CLEAR THE JUNK
2752  010170  021627  000007                    CMP     (SP),#7         ;'CONTROL G' ?
2753  010174  001010                            BNE     1$              ;BR IF NOT
2754  010176  022737  000176  001140            CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER ?
2755  010204  001004                            BNE     1$              ;BR IF NOT ACTIVE
2756  010206  012766  010420  000002            MOV     #KSR2,2(SP)     ;CHANGE THE RETURN ADDRESS
2757  010214  000403                            BR      2$              ;CONTINUE
2758  010216  012737  017026  000060    1$:     MOV     #$TKSRV,TKVEC   ;CHANGE KEYBOARD VECTOR ADDRESS
2759  010224  000137  017032            2$:     JMP     $TKSRV+4        ;PROCESS THE CHARACTER JUST ENTERED
2760  010230  104411                    KSR1:   RDLIN                   ;READ THE KEYBOARD
2761  010232  012737  010146  000060            MOV     #KSR,TKVEC      ;CHANGE THE KEYBOARD VECTOR
2762  010240  012605                            MOV     (SP)+,R5        ;BUFFER ADDRESS
2763  010242  005205                            INC     R5              ;POINT TO DRIVE NUMBER
2764  010244  104401  001213                    TYPE    ,$CRLF          ;CR-LF
2765  010250  122715  000101                    CMPB    #'A,(R5)        ;EQ TO AN 'A'
2766  010254  001410                            BEQ     1$              ;BR IF IT IS
```

```
2767  010256  121527  000067            CMPB    (R5),#67        ;UNIT NUMBER GREATER THAN AN ASCII 7 ?
2768  010262  101054                    BHI     6$              ;BR IF IT IS
2769  010264  121527  000060            CMPB    (R5),#60        ;UNIT NUMBER LESS THAN AN ASCII 0 ?
2770  010270  103451                    BLO     6$              ;BR IF IT IS
2771  010272  142715  177770            BICB    #^C7,(R5)       ;LEAVE ONLY LOWER 3 BITS IF CHAR NOT 'A'
2772  010276  122765  000124  177777 1$:CMPB    #'T,-1(R5)      ;EQ TO 'T'
2773  010304  001003                    BNE     2$              ;BR IF NOT EQ
2774  010306  004737  010424            JSR     PC,NEWASN       ;ASSIGN UNIT FOR TEST
2775  010312  000442                    BR      KSR2            ;EXIT
2776  010314  122765  000104  177777 2$:CMPB    #'D,-1(R5)      ;EQ TO 'D' ?
2777  010322  001003                    BNE     3$              ;BR IF NOT EQ
2778  010324  004737  010664            JSR     PC,DEASGN       ;DEASSIGN UNIT
2779  010330  000433                    BR      KSR2            ;EXIT
2780  010332  122765  000123  177777 3$:CMPB    #'S,-1(R5)      ;EQ TO 'S'
2781  010340  001003                    BNE     4$              ;BR IF NOT EQ
2782  010342  004737  010772            JSR     PC,SCMND        ;TYPE STATISTICS
2783  010346  000424                    BR      KSR2            ;EXIT
2784  010350  122765  000127  177777 4$:CMPB    #'W,-1(R5)      ;EQ TO 'W'
2785  010356  001007                    BNE     5$              ;BR IF NOT EQ
2786  010360  032777  000001  170552    BIT     #SWD,@SWR       ;IS SWITCH 0 SET ?
2787  010366  001012                    BNE     6$              ;BR IF SET, CAN'T DO 'W' COMMAND
2788  010370  004737  011074            JSR     PC,DATAPK       ;WRITE A DATA PACK
2789  010374  000411                    BR      KSR2            ;EXIT
2790  010376  122765  000122  177777 5$:CMPB    #'R,-1(R5)      ;EQ TO 'R' ?
2791  010404  001003                    BNE     6$              ;BR IF NOT EQ
2792  010406  004737  011106            JSR     PC,REDAPK       ;READ A DATA PACK
2793  010412  000402                    BR      KSR2            ;EXIT
2794  010414  104401  033146       6$:  TYPE    ,INVLD          ;TYPE 'INVALID COMMAND' MESSAGE
2795  010420  104413            KSR2:   RESREG                  ;RESTORE R0-R5
2796  010422  000002                    RTI                     ;RETURN
2797
2798                                    ;ROUTINE TO ASSIGN A DRIVE ('T' COMMAND)
2799
2800  010424  005037  001252    NEWASN: CLR     PACK            ;CLEAR 'W' COMMAND INDICATOR
2801  010430  005004            ASGN0:  CLR     R4              ;R4 IS TABLE INDEX
2802  010432  012703  000010            MOV     #8.,R3          ;SET INITIAL COUNT TO 8
2803  010436  122715  000101            CMPB    #101,(R5)       ;ASSIGN ALL UNITS?
2804  010442  001417                    BEQ     ASGN2           ;BR IF ALL UNITS
2805  010444  111504            ASGN1:  MOVB    (R5),R4         ;PUT UNIT # IN R4
2806  010446  012737  032567  011140    MOV     #UNTASN,ASNMSG  ;'UNIT ASSIGNED' MESSAGE ADDR
2807  010454  012703  000001            MOV     #1,R3           ;RELOAD R3 FOR 1 UNIT
2808  010460  136437  021304  025460    BITB    ATABIT(R4),ASNLST  ;UNIT ALREADY ASSIGNED ?
2809  010466  001002                    BNE     1$              ;BR IF IT IS
2810  010470  004737  010524            JSR     PC,ASGN3        ;SEE IF UNIT ON THE SYSTEM
2811  010474  000137  011120       1$:  JMP     ASNERR          ;RETURN HERE IF DRIVE NOT AVAIL
2812  010500  000207                    RTS     PC              ;EXIT
2813  010502  004737  010524    ASGN2:  JSR     PC,ASGN3        ;ASSIGN ALL UNASSIGNED, AVAIL UNITS
2814  010506  000137  010514            JMP     1$              ;UNIT NOT ON SYSTEM
2815  010512  000207                    RTS     PC              ;EXIT
2816  010514  012746  010506       1$:  MOV     #ASGN2+4,-(SP)  ;PUT RETURN ADDRESS ON THE STACK
2817  010520  000137  010626            JMP     ASGN4           ;LOOK FOR MORE UNITS
2818  010524  136437  021304  025460 ASGN3: BITB ATABIT(R4),ASNLST  ;UNIT ALREADY ASSIGNED ?
2819  010532  001035                    BNE     ASGN4           ;BR IF IT IS
2820  010534  005737  021124       1$:  TST     TRNSWT          ;DATA TRANSFER UNDER WAY ?
2821  010540  001375                    BNE     1$              ;BR IF IT IS
2822  010542  110437  025414            MOVB    R4,GENDPB       ;DRIVE NUMBER
```

```
2823  010546  004737  015354                  JSR     PC,RECALO       ;RECALIBRATE DRIVE
2824  010552  105764  021112                  TSTB    DRVSTA(R4)      ;DRIVE AVAILABLE?
2825  010556  001432                          BEQ     ASGN6           ;BR IF DRIVE OFFLINE
2826  010560  100435                          BMI     ASGN7           ;BR DRIVE NOT AVAILABLE
2827  010562  006304                          ASL     R4              ;MAKE R4 INTO WORD INDEX
2828  010564  016464  025736  025526          MOV     BLKADR(R4),NEWUNT(R4) ;DPB ADDRESS
2829  010572  016400  025736                  MOV     BLKADR(R4),R0   ;PUT BLOCK'S ADDR INTO R0
2830  010576  004737  011150                  JSR     PC,CLRDPB       ;CLEAR BLOCK FOR UNIT JUST ASSIGNED
2831  010602  012760  000001  000062          MOV     #1,$PASSC(R0)   ;PRESET PASS COUNT TO 1
2832  010610  005737  001252                  TST     PACK            ;WRITE DATA PACK ?
2833  010614  001403                          BEQ     2$              ;BR IF NOT
2834  010616  113760  001252  000022          MOVB    PACK,$PACK(R0)  ;SET READ/WRITE DATA PACK INDICATOR
2835  010624  006204                  2$:     ASR     R4              ;RESTORE UNIT ADDRESS
2836  010626  005303                  ASGN4:  DEC     R3              ;DEC UNIT COUNT
2837  010630  001402                          BEQ     ASGN5           ;BR IF FINISHED
2838  010632  005204                          INC     R4              ;INCREMENT UNIT NUMBER
2839  010634  000733                          BR      ASGN3           ;CONTINUE
2840  010636  062716  000004          ASGN5:  ADD     #4,(SP)         ;INCREMENT RETURN
2841  010642  000407                          BR      ASGN8           ;EXIT
2842  010644  012737  032524  011140  ASGN6:  MOV     #UNTOFF,ASNMSG  ;'OFFLINE' MESSAGE ADDRESS
2843  010652  000403                          BR      ASGN8           ;EXIT
2844  010654  012737  032615  011140  ASGN7:  MOV     #NOTSAF,ASNMSG  ;ADDR OF 'DRIVE UNSAFE' MESSAGE
2845  010662  000207                  ASGN8:  RTS     PC              ;RETURN
2846
2847                                  ;ROUTINE TO DEASSIGN A UNIT ('D' COMMAND)
2848
2849  010664  005004                  DEASGN: CLR     R4
2850  010666  122715  000101                  CMPB    #101,(R5)       ;DEASSIGN ALL UNITS ?
2851  010672  001434                          BEQ     5$              ;BR IF YES
2852  010674  012703  000001                  MOV     #1,R3           ;SET R3 FOR ONE UNIT
2853  010700  111504                          MOVB    (R5),R4         ;GET UNIT NUMBER
2854  010702  136437  021304  025460  1$:     BITB    ATABIT(R4),ASNLST ;UNIT ASSIGNED ?
2855  010710  001414                          BEQ     3$              ;BR IF NOT
2856  010712  146437  021304  025460          BICB    ATABIT(R4),ASNLST ;DELETE THE UNIT FROM THE ASSIGNED LIST
2857  010720  006304                          ASL     R4              ;MAKE ADDR INTO A WORD INDEX
2858  010722  016464  025736  025504          MOV     BLKADR(R4),DUNIT(R4) ;PUT ADDRESS IN DEASSIGN LIST
2859  010730  006204                          ASR     R4
2860  010732  005303                  2$:     DEC     R3              ;ANY MORE UNITS ?
2861  010734  001412                          BEQ     4$              ;BR IF NOT
2862  010736  005204                          INC     R4
2863  010740  000760                          BR      1$
2864  010742  122715  000101          3$:     CMPB    #101,(R5)       ;DEASSIGN ALL UNITS ?
2865  010746  001771                          BEQ     2$              ;BR IF YES
2866  010750  012737  032545  011140          MOV     #UNNOT,ASNMSG   ;ADDR OF 'NOT ASSIGNED' MESSAGE
2867  010756  004737  011120                  JSR     PC,ASNERR       ;REPORT IT
2868  010762  000207                  4$:     RTS     PC
2869  010764  012703  000010          5$:     MOV     #8.,R3          ;SET UNIT COUNT TO 8
2870  010770  000744                          BR      1$
2871
2872                                  ;ROUTINE TO TYPE UNIT PERFORMANCE SUMMARY  ('S' COMMAND)
2873
2874  010772  005004                  SCMND:  CLR     R4
2875  010774  122715  000101                  CMPB    #101,(R5)       ;ALL STATISTICS ?
2876  011000  001421                          BEQ     2$              ;BR IF YES
2877  011002  111504                          MOVB    (R5),R4         ;GET UNIT #
2878  011004  136437  021304  025460          BITB    ATABIT(R4),ASNLST ;SEE IF UNIT ASSIGNED
```

```
2879  011012  001406                          BEQ     1$              ;BR IF NOT
2880  011014  006304                          ASL     R4              ;MAKE UNIT ADDR INTO WORD INDEX
2881  011016  016400  025736                  MOV     BLKADR(R4),R0   ;ADDR OF BLOCK
2882  011022  004737  007240                  JSR     PC,TYPEST       ;TYPE DRIVE STATISTICS
2883  011026  000421                          BR      5$              ;EXIT
2884  011030  012737  032545  011140  1$:     MOV     #UNTNOT,ASNMSG  ;ADDR OF 'NOT ASSIGNED' MSG
2885  011036  004737  011120                  JSR     PC,ASNERR       ;TYPE ERROR MESSAGE
2886  011042  000413                          BR      5$              ;EXIT
2887  011044  012703  000010          2$:     MOV     #8.,R3          ;UNIT COUNT
2888  011050  105737  025460          3$:     TSTB    ASNLST          ;SEE IF ANY UNIT ASSIGNED
2889  011054  001004                          BNE     4$              ;BR IF ANY ARE
2890  011056  005204                          INC     R4              ;INCREMENT UNIT ADDRESS
2891  011060  005303                          DEC     R3              ;DECREMENT COUNTER
2892  011062  001372                          BNE     3$              ;MORE TO CHECK ?
2893  011064  000402                          BR      5$              ;NONE ASSIGNED, RETURN
2894  011066  004737  007126          4$:     JSR     PC,STATPR       ;TYPE ALL STATISTICS
2895  011072  000207                  5$:     RTS     PC
2896
2897                                  ;ROUTINE TO WRITE A DATA PACK ('W' COMMAND)
2898
2899  011074  012737  177777  001252  DATAPK: MOV     #-1,PACK        ;SET THE 'W' COMMAND INDICATOR
2900  011102  000137  010430                  JMP     ASGNO           ;ASSIGN REQUESTED UNIT
2901
2902
2903                                  ;ROUTINE TO READ A DATA PACK ('R' COMMAND)
2904
2905  011106  012737  000001  001252  REDAPK: MOV     #1,PACK         ;SET THE 'READ' INDICATOR
2906  011114  000137  010430                  JMP     ASGNO           ;ASSIGN THE REQUESTED UNIT
2907
2908                                  ;TYPEOUT ASSIGN/DEASSIGN ERROR MESSAGE
2909
2910  011120  104401  033144          ASNERR: TYPE    ,QUES           ;QUESTION MARK
2911  011124  104401  032515                  TYPE    ,UNTMSG         ;TYPE 'UNIT'
2912  011130  010446                          MOV     R4,-(SP)        ;DRIVE NUMBER
2913  011132  104403                          TYPOS                   ;TYPE IT
2914  011134      001      000                .BYTE   1,0
2915  011136  104401                          TYPE                    ;TYPE SPECIFIC MESSAGE
2916  011140  000000          ASNMSG: .WORD   0                       ;MESSAGE ADDRESS
2917  011142  104401  001213                  TYPE    ,$CRLF
2918  011146  000207                          RTS     PC
2919
2920                                  ;ROUTINE TO CLEAR THE DPB FOR THE ASSIGNED UNIT
2921
2922  011150                          CLRDPB:
2923  011150  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2924  011152  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2925  011154  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2926  011156  010004                          MOV     R0,R4           ;GET THE DPB ADDRESS
2927  011160  062704  000002                  ADD     #2,R4           ;ADDRESS OF FIRST LOCN TO BE CLEARED
2928  011164  012703  000005                  MOV     #5,R3           ;NUMBER OF LOCNS TO BE CLEARED
2929  011170  005024          1$:     CLR     (R4)+           ;CLEAR THE LOCATION
2930  011172  005303                          DEC     R3              ;DECREMENT THE COUNTER
2931  011174  001375                          BNE     1$              ;BR IF NOT FINISHED
2932  011176  062704  000002                  ADD     #2,R4           ;MOVE THE ADDRESS PAST THE 'REG' ADDR
2933  011202  012703  000074                  MOV     #$$ILO-$REG,R3  ;NUMBER OF LOCNS TO BE CLEARED
2934  011206  005024          2$:     CLR     (R4)+           ;CLEAR
```

```
2935  011210  162703  000002          SUB     #2,R3           ;DECREMENT THE LOCN COUNTER
2936  011214  001374                  BNE     2$              ;BR IF NOT FINISHED
2937  011216  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
2938  011220  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
2939  011222  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
2940  011224  000207                  RTS     PC
2941
2942                          ;DEASSIGN A DRIVE IF A FATAL ERROR OCCURS
2943
2944  011226  011446          DROP:   MOV     (R4),-(SP)      ;SAVE R4
2945  011230  005004                  CLR     R4              ;CLEAR R4 FOR UNIT NUMBER
2946  011232  111004                  MOVB    (R0),R4         ;MOVE UNIT NUMBER TO R4
2947  011234  146437  021304  025460  BICB    ATABIT(R4),ASMLST  ;REMOVE UNIT FROM ASSIGNED LIST
2948  011242  006304                  ASL     R4              ;MAKE UNIT NUMBER INTO A TABLE INDEX
2949  011244  010064  025504          MOV     R0,DUNIT(R4)    ;PUT UNIT IN DROP LIST
2950  011250  104401  001213          TYPE    ,$CRLF          ;CR-LF
2951  011254  104401  001213          TYPE    ,$CRLF          ;CR-LF
2952  011260  104401  033001          TYPE    ,DROPNG         ;TYPE 'DROPPING UNIT'
2953  011264  104401  033103          TYPE    ,DRNUM          ;'DRIVE #'
2954  011270  111046                  MOVB    (R0),-(SP)      ;DRIVE NUMBER
2955  011272  104403                  TYPOS                   ;TYPE IT
2956  011274     001     000          .BYTE   1,0
2957  011276  104401  001213          TYPE    ,$CRLF          ;CR-LF
2958  011302  012604                  MOV     (SP)+,R4        ;RESTORE R4
2959  011304  000207                  RTS     PC              ;RETURN
2960
2961                          ;ROUTINE TO DEASSIGN A UNIT IF ERRORS BECOMES EXCESSIVE
2962
2963  011306  032777  000020  167624  ABNRML: BIT     #SW04,@SWR      ;SEE IF SWITCH 4 SET
2964  011314  001006                  BNE     1$              ;BR IF IT'S SET
2965  011316  023760  001310  000050          CMP     MAXER,$TOTAL(R0)  ;CHECK TOTAL ERROR VALUE
2966  011324  101002                  BHI     1$              ;BR IF NOT AT MAXIMUM
2967  011326  000137  011226          JMP     DROP            ;DEASSIGN THE DRIVE
2968  011332  000207                  1$:     RTS     PC              ;RETURN
2969
2970                          ;ROUTINE TO CHECK FOR END OF PASS AND END OF TEST
2971
2972  011334  005737  001316          NRML:   TST     SEKMOD          ;END OF PASS DETERMINED BY SEEKS OR WORDS ?
2973  011340  001012                  BNE     NRML1           ;BR IF SEEKS
2974  011342  026037  000046  001276          CMP     $READ+2(R0),ENDCON+2  ;CHECK MSW OF WORDS READ COUNT
2975  011350  101017                  BHI     NRML2           ;BR IF MSW GREATER THAN LIMIT
2976  011352  103405                  BLO     NRML1           ;BR IF MSW LESS THAN LIMIT
2977  011354  026037  000044  001274          CMP     $READ(R0),ENDCON  ;CHECK LSW AGAINST LIMIT
2978  011362  103012                  BHIS    NRML2           ;BR IF EQUAL OR GREATER
2979  011364  000504                  BR      NRMLX           ;EXIT
2980  011366  026037  000042  001302  NRML1:  CMP     $POSIT+2(R0),ENDSEK+2  ;CHECK MSW OF SEEK COUNT
2981  011374  101005                  BHI     NRML2           ;BR IF MSW GREATER THAN LIMIT
2982  011376  103477                  BLO     NRMLX           ;EXIT IF MSW LESS THAN LIMIT
2983  011400  026037  000040  001300          CMP     $POSIT(R0),ENDSEK  ;CHECK LSW OF SEEK COUNT
2984  011406  103473                  BLO     NRMLX           ;EXIT IF LSW LESS THAN LIMIT
2985  011410  104401  001213          NRML2:  TYPE    ,$CRLF          ;CR-LF
2986  011414  104401  033035                  TYPE    ,ENDPAS         ;END OF PASS FOR THE UNIT
2987  011420  016046  000062                  MOV     $PASSC(R0),-(SP)  ;PUT PASS COUNT ON THE STACK
2988  011424  104405                  TYPOS                   ;CONVERT PASS COUNT TO DECIMAL AND TYPE IT
2989  011426  032777  000020  167504          BIT     #SW04,@SWR      ;SWITCH 4 SET ?
2990  011434  001016                  BNE     1$              ;BR IF SET
```

```
 2991  011436  026037  000062  001304          CMP     $PASSC(R0),PASCNT  ;SEE IF AT END OF TEST
 2992  011444  103412                          BLO     1$                 ;BR IF NOT
 2993  011446  104401  033051                  TYPE    ,ENDTST            ;TYPE 'END OF TEST'
 2994  011452  104401  033103                  TYPE    ,DRNUM             ;'DRIVE #'
 2995  011456  111046                          MOVB    (R0),-(SP)         ;DRIVE NUMBER
 2996  011460  104403                          TYPOS                      ;TYPE IT
 2997  011462      001      000                .BYTE   1,0
 2998  011464  104401  001213                  TYPE    ,$CRLF             ;CR-LF
 2999  011470  000430                          BR      3$                 ;DEASSIGN THE DRIVE
 3000  011472  104401  033103          1$:     TYPE    ,DRNUM             ;'DRIVE #'
 3001  011476  111046                          MOVB    (R0),-(SP)         ;DRIVE NUMBER
 3002  011500  104403                          TYPOS                      ;TYPE IT
 3003  011502      001      000                .BYTE   1,0
 3004  011504  104401  001213                  TYPE    ,$CRLF             ;CR-LF
 3005  011510  004737  007240                  JSR     PC,TYPEST          ;TYPE THE UNIT'S STATISTICS
 3006  011514  010346                          MOV     R3,-(SP)           ;SAVE R3
 3007  011516  010446                          MOV     R4,-(SP)           ;SAVE R4
 3008  011520  010004                          MOV     R0,R4              ;UNIT'S BLOCK ADDRESS
 3009  011522  062704  000034                  ADD     #$OPERC,R4         ;ADD THE STARTING ADDR OF SECTIONS TO CLEAR
 3010  011526  012703  000006                  MOV     #6,R3              ;NUMBER OF LOCNS TO BE CLEARED
 3011                                                                     ;(ERROR COUNTERS NOT CLEARED)
 3012  011532  005024                  2$:     CLR     (R4)+              ;CLEAR THE LOCN
 3013  011534  005303                          DEC     R3                 ;DECREMENT THE LOCATION COUNTER
 3014  011536  001375                          BNE     2$                 ;BR IF MORE TO GO
 3015  011540  012604                          MOV     (SP)+,R4           ;RESTORE R4
 3016  011542  012603                          MOV     (SP)+,R3           ;RESTORE R3
 3017  011544  005260  000062                  INC     $PASSC(R0)         ;INCREMENT THE PASS COUNT
 3018  011550  000412                          BR      NRMLX              ;EXIT
 3019  011552  104401  001213          3$:     TYPE    ,$CRLF
 3020  011556  005004                          CLR     R4                 ;CLEAR R4 FOR DRIVE NUMBER
 3021  011560  111004                          MOVB    (R0),R4            ;MOVE DRIVE NUMBER
 3022  011562  146437  021304  025460          BICB    ATABIT(R4),ASNLST  ;DELETE DRIVE FROM ASSIGNED LIST
 3023  011570  006304                          ASL     R4                 ;MAKE DRIVE NUMBER INTO TABLE INDEX
 3024  011572  010064  025504                  MOV     R0,DUNIT(R4)       ;PUT BLOCK ADDRESS INTO DROP LIST
 3025  011576  000207                  NRMLX:  RTS     PC                 ;RETURN
 3026
 3027                                  ;ROUTINE TO GET THE REMAINDER OF THE RANDOM NUMBER
 3028
 3029  011600  013746  020620          GETREM: MOV     $HINUM,-(SP)       ;LOWER DIVIDEND
 3030  011604  005046                          CLR     -(SP)              ;UPPER DIVIDENT
 3031  011606  010546                          MOV     R5,-(SP)           ;PUT DIVISOR ON THE STACK
 3032  011610  004737  020206                  JSR     PC,$DIV
 3033  011614  012605                          MOV     (SP)+,R5           ;PUT THE REMAINDER INTO R5
 3034  011616  005726                          TST     (SP)+              ;ADJUST THE STACK POINTER
 3035  011620  000207                          RTS     PC
 3036
 3037                                  ;ROUTINE TO REPLACE LEADING ZEROS IN A NUMERIC STRING WITH SPACES
 3038
 3039  011622  012737  000004  011742  SRPZR8: MOV     #4,RPZRC           ;SETUP TO TYPE 8 SIGNIFICANT DIGITS
 3040  011630  000412                          BR      RPZERO             ;TYPE THE VALUE
 3041  011632  012737  000006  011742  SRPZR4: MOV     #6,RPZRC           ;SETUP TO TYPE 4 SIGNIFICANT DIGITS
 3042  011640  000406                          BR      RPZERO             ;TYPE THE VALUE
 3043  011642  012737  000007  011742  SRPZR5: MOV     #7,RPZRC           ;SETUP TO TYPE 3 SIGNIFICANT DIGITS
 3044  011650  000402                          BR      RPZERO             ;TYPE THE VALUE
 3045  011652  005037  011742          SRPZR0: CLR     RPZRC              ;SETUP TO TYPE 10 SIGNIFICANT DIGITS
 3046  011656  010046                  RPZERO: MOV     R0,-(SP)           ;SAVE R0
```

```
                                                MOV     4(SP),R0        ;ADDRESS OF NUMBER TO R0
                                        1$:     CMPB    #'0,(R0)        ;BYTE EQUAL TO ASCII '0' ?
                                                BNE     2$              ;BR IF NOT
                                                MOVB    #40,(R0)        ;REPLACE THE ZERO WITH A SPACE
                                                INC     R0              ;INCREMENT THE BYTE ADDRESS
                                                BR      1$              ;GO BACK AND LOOK FOR MORE LEADING ZEROS
                                        2$:     TSTB    (R0)            ;SEE IF ZERO BYTE TERMINATOR
                                                BNE     3$              ;BR IF NOT
                                                DEC     R0              ;BACKUP STRING POINTER
                                                MOVB    #'0,(R0)        ;PUT A ZERO BACK IN
                                        3$:     MOV     4(SP),4$        ;PUT ADDRESS IN LOCATION FOR TYPEOUT
                                                ADD     RPZRC,4$        ;BEGINNING OF SIGNIFICANT DIGITS
                                                TYPE                    ;TYPE THE NUMBER
                                        4$:     .WORD   0               ;ADDRESS OF NUMBER
                                                MOV     (SP)+,R0        ;RESTORE R0
                                                MOV     (SP)+,(SP)      ;MOVE RETURN ADDRESS
                                                RTS     PC              ;RETURN

011742  000000                          RPZRC:  .WORD   0               ;OFFSET FOR CONVERTED DIGITS HERE

                                        ;THIS ROUTINE IS USED TO CHECK IF AN ASCII CHARACTER IS A DIGIT
                                        ;BETWEEN 0 AND 7.
                                        ;
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R0,CK.OCT       ;CHECK THE CHARACTER
                                        ;       RETURN1                 ;CHARACTER IS IN R2 AS A
                                        ;                               ;OCTAL DIGIT
                                        ;       RETURN2                 ;CHARACTER IS NOT BETWEEN 0-7

011744  121127  000060                 CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
011750  103407                                 BLO     1$              ;YES -- BRANCH
011752  121127  000067                         CMPB    (R1),#'7        ;GREATER THAN SEVEN?
011756  101004                                 BHI     1$              ;YES -- BRANCH
011760  111102                                 MOVB    (R1),R2         ;GET THE CHARACTER
011762  042702  177770                         BIC     #^C7,R2         ;STRIP AWAY THE ASCII
011766  000401                                 BR      2$              ;BYPASS RETURN ADJUST
011770  005720                         1$:     TST     (R0)+           ;ADJUST FOR RETURN
011772  000200                         2$:     RTS     R0              ;RETURN

                                        ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
                                        ;DETERMINE WHAT IT IS.
                                        ;
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R0,CK.CHR       ;CHECK CHARACTER
                                        ;       RETURN  ADR1            ;UNKNOWN CHARACTER
                                        ;       RETURN  ADR2            ;CARRIAGE RETURN # (R1)=ADR+1
                                        ;       RETURN  ADR3            ;PERIOD # (R1)=ADR+1

011774  105711                         CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
011776  001404                                 BEQ     1$              ;YES -- BRANCH
012000  121127  000056                         CMPB    (R1),#'.        ;"PERIOD"?
012004  001003                                 BNE     2$              ;NO -- BRANCH
012006  005720                                 TST     (R0)+           ;PERIOD
012010  005720                         1$:     TST     (R0)+           ;CARRIAGE RETURN
012012  005201                                 INC     R1              ;MOVE POINTER TO NEXT CHARACTER
012014  011000                         2$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
```

```
 3103  012016  000200                        RTS     R0              ;RETURN
 3104
 3105
 3106                                  ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
 3107                                  ;AND FORMS AN OCTAL NUMBER IN R2
 3108                                  ;
 3109                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
 3110                                  ;       JSR     R0,CK.NUM       ;GO FORM THE NUMBER
 3111                                  ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
 3112                                  ;       RETURN  ADR2            ;"CR"--R2=NUMBER
 3113                                  ;       RETURN  ADR3            ;"PERIOD"--R2=NUMBER
 3114                                  ;       RETURN  ADR4            ;'CR' WAS FIRST ENTRY
 3115                                  ;       RETURN  ADR5            ;'PERIOD' WAS FIRST ENTRY
 3116
 3117  012020  010346                  CK.NUM: MOV     R3,-(SP)        ;SAVE R3
 3118  012022  005003                          CLR     R3              ;START NUMBER AT ZERO
 3119  012024  004037  011744                  JSR     R0,CK.OCT       ;OCTAL DIGIT?
 3120  012030  000405                          BR      1$              ;YES--BRANCH
 3121  012032  004037  011774                  JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
 3122  012036  012122                          6$                      ;ILLEGAL CHARACTER
 3123  012040  012112                          8$                      ;CARRIAGE RETURN
 3124  012042  012110                          7$                      ;"."
 3125  012044  005201                  1$:     INC     R1              ;MOVE TO NEXT CHARACTER
 3126  012046  006303                          ASL     R3              ;FOR THE OCTAL NUMBER IN R3
 3127  012050  103424                          BCS     6$              ;DON'T LET IT GET TO BIG
 3128  012052  006303                          ASL     R3
 3129  012054  103422                          BCS     6$
 3130  012056  006303                          ASL     R3
 3131  012060  103420                          BCS     6$
 3132  012062  060203                          ADD     R2,R3
 3133  012064  004037  011744                  JSR     R0,CK.OCT       ;IS THIS AN OCTAL DIGIT?
 3134  012070  000765                          BR      1$              ;YES--MAKE IT PART OF THE NUMBER
 3135  012072  010302                  2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
 3136  012074  005003                          CLR     R3              ;START WITH ZERO INDEX
 3137  012076                          3$:
 3138  012076  004037  011774                  JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
 3139  012102  012122                          6$                      ;ILLEGAL CHARACTER
 3140  012104  012116                          5$                      ;CARRIAGE RETURN
 3141  012106  012114                          4$                      ;"."
 3142  012110  005723                  7$:     TST     (R3)+           ;'PERIOD' ONLY
 3143  012112  005723                  8$:     TST     (R3)+           ;'CR' ONLY
 3144  012114  005723                  4$:     TST     (R3)+           ;"PERIOD"
 3145  012116  005723                  5$:     TST     (R3)+           ;"CR"
 3146  012120  060300                          ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
 3147  012122  012603                  6$:     MOV     (SP)+,R3        ;RESTORE R3
 3148  012124  011000                          MOV     (R0),R0         ;PICKUP EXIT ADDRESS
 3149  012126  000200                          RTS     R0              ;RETURN
 3150
 3151                                  ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
 3152                                  ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
 3153                                  ;TABLE" (SERRTB), AND REPORTS THE APPROPIATE INFORMATION
 3154                                  ;CONCERNING THE ERROR.
 3155
 3156  012130  104412                  TYPERR: SAVREG                  ;SAVE R0-R5
 3157  012132  032777  020000  167000          BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS ?
 3158  012140  001113                          BNE     20$             ;BR IF INHIBIT
```

```
3159  012142  005000                        CLR     R0              ;CLEAR R0 FOR ERROR NUMBER
3160  012144  113700  001114                MOVB    $ITEMB,R0       ;ERROR NUMBER
3161  012150  005300                        DEC     R0              ;FORM INDEX FOR ERROR TABLE
3162  012152  006300                        ASL     R0
3163  012154  006300                        ASL     R0
3164  012156  006300                        ASL     R0
3165  012160  062700  001346        1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
3166  012164  012037  012204                MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
3167  012170  001406                        BEQ     3$              ;BRANCH IF THERE ISN'T ONE
3168  012172  104401  001213                TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
3169  012176  004737  007634                JSR     PC,$TIME        ;TYPE THE TIME
3170  012202  104401                        TYPE
3171  012204  000000        2$:     .WORD   0                       ;"EM" POINTER GOES HERE
3172  012206  012037  012222        3$:     MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
3173  012212  001404                        BEQ     5$              ;BRANCH IF NONE
3174  012214  104401  001213                TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3175  012220  104401                        TYPE
3176  012222  000000        4$:     .WORD   0                       ;"DH" POINTER GOES HERE
3177  012224  012001        5$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
3178  012226  001460                        BEQ     20$             ;BRANCH IF NONE
3179  012230  005005                        CLR     R5              ;SET INDENT SWITCH
3180  012232  012000                        MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
3181  012234  012002                        MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
3182  012236  001451                        BEQ     17$             ;BRANCH IF DH NUMBER IS 0
3183  012240  005105                        COM     R5              ;NO INDENT
3184  012242  104401  001213                TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3185  012246  112003        10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
3186  012250  112004                        MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
3187  012252  006004        11$:    ROR     R4              ;OCTAL OR DECIMAL?
3188  012254  103403                        BCS     12$             ;DECIMAL--BRANCH
3189  012256  013146                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3190  012260  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3191  012262  000402                        BR      13$
3192  012264                        12$:
3193  012264  013146                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3194  012266  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3195  012270  005303        13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
3196  012272  001403                        BEQ     14$             ;NO--BRANCH
3197  012274  104401  032500                TYPE    ,SPACE2         ;YES--TYPE SEPERATORS
3198  012300  000764                        BR      11$             ;LOOP
3199  012302  005302        14$:    DEC     R2              ;MORE DH'S?
3200  012304  003431                        BLE     20$             ;NO--BRANCH
3201  012306  104401  001213                TYPE    ,$CRLF          ;YES--START A NEW LINE
3202  012312  005760  000002                TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
3203  012316  001404                        BEQ     15$             ;BR IF YES - BYPASS THE INDENT
3204  012320  005105                        COM     R5              ;INDENT?
3205  012322  001002                        BNE     15$             ;NO--BRANCH
3206  012324  104401  032500                TYPE    ,SPACE2         ;YES--TYPE SPACES
3207  012330  012037  012336        15$:    MOV     (R0)+,16$       ;GET NEXT DH
3208  012334  104401                        TYPE                    ;AND TYPE IT
3209  012336  000000        16$:    .WORD   0                       ;DH POINTER GOES HERE
3210  012340  005710                        TST     (R0)            ;TYPE A 'DT' ?
3211  012342  001003                        BNE     21$             ;BR IF A 'DT'
3212  012344  062700  000004                ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
3213  012350  000754                        BR      14$             ;SEE IF END OF 'DF' BLOCK
3214  012352  104401  001213        21$:    TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
```

```
3215  012356  005705                        TST     R5              ;INDENT?
3216  012360  001332                        BNE     10$             ;NO--BRANCH
3217  012362  104401  032500        17$:    TYPE    SPACE2          ;YES--TYPE SPACES
3218  012366  000727                        BR      10$             ;LOOP
3219  012370  104413                20$:    RESREG                  ;RESTORE R0-R5
3220  012372  000207                        RTS     PC              ;RETURN
3221
3222                                ;;*****************************************************************
3223
3224                                .SBTTL  END OF ORDER PROCESSING
3225
3226                                ;;*****************************************************************
3227
3228                                ;PROCESS THE ORDER TERMINATION
3229
3230  012374  005760  000016        PROCES: TST     STATUS(R0)      ;SEE IF RP11 HANDLER SIGNALED AN ERROR
3231  012400  100002                        BPL     1$              ;BR IF NO ERROR DETECTED
3232  012402  000137  013462                JMP     ERPROC          ;ERROR - PROCESS IT
3233  012406  004737  012434        1$:     JSR     PC,CKERR        ;NO ERROR, CHECK ERROR BITS ANYWAY
3234  012412  004737  012466                JSR     PC,CKBUS        ;NO ERROR, CHECK BUS ADDR & WC
3235  012416  032777  000002  166514        BIT     #SWD1,@SWR      ;COMPARE THE BUFFER ?
3236  012424  001002                        BNE     2$              ;BR IF NOT
3237  012426  004737  012534                JSR     PC,CMPAR        ;NO ERROR, COMPARE DATA
3238  012432  000207                2$:     RTS     PC              ;RETURN
3239
3240                                ;CHECK ERROR BITS IN 'RPER' & 'RPDS'
3241
3242  012434  032760  015000  000066  CKERR:  BIT     #HNF!SUSI!SUFU,$RPDS(R0)  ;SEE IF ERROR BITS IN RPDS
3243  012442  001003                        BNE     1$              ;BR IF ANY SET
3244  012444  005760  000070                TST     $RPER(R0)       ;ANY BITS SET IN RPER
3245  012450  001405                        BEQ     2$              ;BR IF NONE SET
3246  012452  104015                1$:     ERROR   15              ;ERROR BITS SET, NO ERROR SIGNALED
3247  012454  004737  007616                JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
3248  012460  062706  000002                ADD     #2,SP           ;RETURN TO MAIN ROUTINE
3249  012464  000207                2$:     RTS     PC              ;RETURN
```

```
                                        ;CHECK BUS ADDRESS REGISTER & WORD COUNT REGISTER
                                        ;
012466 016046 000020            CKBUS:  MOV     $WRDL(R0),-(SP) ;WORD LENGTH
012472 006316                           ASL     (SP)            ;CHANGE INTO BYTE COUNT
012474 066016 000006                    ADD     $BUF(R0),(SP)   ;ADD THE STARTING LOCATION
012500 011637 001124                    MOV     (SP),$GDDAT     ;STORE IN CASE OF ERROR
012504 022660 000076                    CMP     (SP)+,$RPBA(R0) ;BUFFER ADDRESS PROPER ?
012510 001003                           BNE     1$              ;BR IF NOT CORRECT
012512 005760 000074                    TST     $RPWC(R0)       ;CHECK WORD COUNT
012516 001405                           BEQ     2$              ;BR IF ZERO
012520 104016                   1$:     ERROR   16              ;WORD COUNT OR BUFFER ADDRESS INCORRECT
012522 004737 007616                    JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012526 062706 000002                    ADD     #2,SP           ;RETURN TO MAIN ROUTINE
012532 000207                   2$:     RTS     PC              ;RETURN
                                        ;
                                        ;COMPARE THE BUFFER
                                        ;
012534 122760 000007 000002     CMPAR:  CMPB    #WRTCHK,$COMND(R0)  ;WRITE CHECK ?
012542 001404                           BEQ     1$              ;BR IF WRITE CHECK
012544 132760 000004 000002             BITB    #BIT02,$COMND(R0)   ;READ ORDER ?
012552 001001                           BNE     CMPARD          ;BR IF READ
012554 000207                   1$:     RTS     PC              ;RETURN
012556 016001 000006            CMPARD: MOV     $BUF(R0),R1     ;BUFFER ADDRESS
012562 016037 000020 013154             MOV     $WRDL(R0),CMCNT  ;WORD COUNT TO WORKING LOCATION
012570 066037 000074 013154             ADD     $RPWC(R0),CMCNT  ;CALCULATE ACTUAL WORDS TRANSFERED
012576 016037 000010 013156             MOV     $CYL(R0),CMCYL  ;CYLINDER ADDRESS
012604 116037 000013 013162             MOVB    $TRK(R0),CMTRK  ;TRACK ADDRESS
012612 116037 000012 013160             MOVB    $SEC(R0),CMSEC  ;SECTOR ADDRESS
012620 013737 001314 013152     CMSTR:  MOV     CMPLMT,LIMIT    ;DISPLAY LIMIT
012626 005237 013152                    INC     LIMIT           ;CONVERT PARAMETER INTO LIMIT VALUE
012632 010137 013164                    MOV     R1,CMBUF        ;STARTING ADDRESS OF SECTOR BUFFER
012636 112737 177777 013142             MOVB    #-1,ZROIND      ;CLEAR THE 'ZERO'S' INDICATOR
012644 105037 013143                    CLRB    FRSTER          ;CLEAR 'FIRST ERROR' INDICATOR
012650 005037 013150                    CLR     ERCTR           ;CLEAR ERROR COUNTER
012654 005037 013144                    CLR     SAVER1          ;CLEAR THE R1 SAVE WORD
012660 005037 013146                    CLR     SAVER5          ;CLEAR THE R5 SAVE WORD
012664 023727 013154 000400             CMP     CMCNT,#256.     ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
012672 101003                           BHI     1$              ;BR IF IT IS
012674 013702 013154                    MOV     CMCNT,R2        ;LESS THAN, USE REMAINING BUFFER
012700 000402                           BR      2$              ;
012702 012702 000400            1$:     MOV     #256.,R2        ;COMPARE SECTOR
012706 160237 013154            2$:     SUB     R2,CMCNT        ;DECREMENT WORD COUNT
012712 004737 013404            CMDAT:  JSR     PC,MATCH        ;FIND THE PATTERN
012716 000403                           BR      2$              ;FOUND A PATTERN
012720 004737 015040                    JSR     PC,NOMTCH       ;RETURN HERE IF NO MATCH WITH PATTERN MADE
012724 000456                           BR      9$              ;BYPASS COMPARE ROUTINE
012726 011405                   2$:     MOV     (R4),R5         ;ADDRESS OF PATTERN ADDRESS IN R4
012730 012703 000020                    MOV     #20,R3          ;R3 IS PATTERN POS COUNTER
012734 022125                   3$:     CMP     (R1)+,(R5)+     ;COMPARE BUFFER WITH PATTERN
012736 001016                           BNE     5$              ;BR IF NOT EQUAL
012740 005737 013150                    TST     ERCTR           ;ERRORS DETECTED ?
012744 001406                           BEQ     4$              ;BR IF NO ERRORS
012746 032777 000010 166164             BIT     #SW3,@SWR       ;SWITCH 3 SET ?
012754 001402                           BEQ     4$              ;BR IF NOT SET
012756 004737 013166                    JSR     PC,CMPRT        ;DISPLAY THE WORD
```

```
3306  012762  005302                    4$:   DEC     R2              ;DECREMENT SIZE COUNT
3307  012764  001434                          BEQ     8$              ;BR WHEN AT END
3308  012766  005303                          DEC     R3              ;DECREMENT PATT POS COUNT
3309  012770  001361                          BNE     3$              ;BR IF NOT AT END OF PATT
3310  012772  000755                          BR      2$              ;RESTART THE PATTERN
3311  012774  005761  177776                5$:   TST     -2(R1)          ;IS MISCOMPARED CHARACTER=0
3312  013000  001406                          BEQ     6$              ;BR IF YES
3313  013002  112737  177777  013142          MOVB    #-1,ZROIND      ;SET NON-ZERO MISCOMPARED IND
3314  013010  004737  013166                  JSR     PC,CMPRT        ;REPORT ERROR
3315  013014  000762                          BR      4$              ;CONTINUE COMPARE
3316  013016  105737  013143                6$:   TSTB    FRSTER          ;FIRST ERROR?
3317  013022  001007                          BNE     7$              ;BR IF NOT
3318  013024  105037  013142                  CLRB    ZROIND          ;SET THE ZERO INDICATOR
3319  013030  010137  013144                  MOV     R1,SAVER1       ;SAVE CURRENT R1
3320  013034  010537  013146                  MOV     R5,SAVER5       ;SAVE CURRENT R5
3321  013040  000750                          BR      4$              ;CONTINUE COMPARE
3322  013042  105737  013142                7$:   TSTB    ZROIND          ;ANY MISCOMPARIONS NOT ZEROS ?
3323  013046  001745                          BEQ     4$              ;BR IF NONE; ALL ERRORS ARE ZERO
3324  013050  004737  013166                  JSR     PC,CMPRT        ;REPORT ERROR
3325  013054  000742                          BR      4$              ;CONTINUE COMPARING
3326  013056  004737  013362                8$:   JSR     PC,ENDCMP       ;PRINT LAST LINE (IF ERRORS)
3327  013062  005737  013154                9$:   TST     CMCNT           ;AT END OF BUFFER
3328  013066  003424                          BLE     CMPRX           ;BR IF AT END
3329  013070  005237  013160                  INC     CMSEC           ;INCREMENT SECTOR
3330  013074  023727  013160  000012          CMP     CMSEC,#10.      ;SECTOR GREATER THAN MAX ?
3331  013102  103646                          BLO     CMSTR           ;BR IF NOT GREATER THAN MAX
3332  013104  005037  013160                  CLR     CMSEC           ;CLEAR SECTOR ADDRESS
3333  013110  005237  013162                  INC     CMTRK           ;INCREMENT TRACK
3334  013114  023727  013162  000024          CMP     CMTRK,#20.      ;TRACK GREATER THAN MAX ?
3335  013122  103636                          BLO     CMSTR           ;BR IF NOT GREATER
3336  013124  005037  013162                  CLR     CMTRK           ;RESET TRACK ADDRESS
3337  013130  005237  013156                  INC     CMCYL           ;INCREMENT CYLINDER ADDRESS
3338  013134  000137  012620                  JMP     CMSTR           ;CONTINUE WITH COMPARE
3339  013140  000207                  CMPRX:  RTS     PC
3340
3341  013142     377                  ZROIND: .BYTE   -1              ;ZERO INDICATOR
3342  013143     000                  FRSTER: .BYTE   0               ;FIRST ERROR INDICATOR
3343                                                                  ;IF NOT 0, MISCOMPARSION FOUND
3344  013144  000000                  SAVER1: .WORD   0               ;SAVE R1 HERE
3345  013146  000000                  SAVER5: .WORD   0               ;SAVE R5 HERE
3346  013150  000000                  ERCTR:  .WORD   0               ;NUMBER OF ERRORS
3347  013152  000000                  LIMIT:  .WORD   0               ;DISPLAY LIMIT
3348  013154  000000                  CMCNT:  .WORD   0               ;WORD COUNT
3349  013156  000000                  CMCYL:  .WORD   0               ;CYLINDER ADDRESS
3350  013160  000000                  CMSEC:  .WORD   0               ;SECTOR ADDRESS
3351  013162  000000                  CMTRK:  .WORD   0               ;TRACK ADDRESS
3352  013164  000000                  CMBUF:  .WORD   0               ;BEGINNING ADDRESS OF SECTOR
3353
3354                                  ;TYPE DATA COMPARE ERRORS
3355
3356  013166  105737  013143          CMPRT:  TSTB    FRSTER          ;FIRST ERROR?
3357  013172  001010                          BNE     1$              ;BR IF NOT
3358  013174  112737  000012  001114          MOVB    #12,$ITEMB      ;ERROR TABLE INDEX
3359  013202  004737  012130                  JSR     PC,TYPERR       ;DATA COMPARISON ERROR
3360  013206  112737  177777  013143          MOVB    #-1,FRSTER      ;SET 'FIRST ERROR' INDICATOR
3361  013214  005737  013144          1$:     TST     SAVER1          ;SAVED REGISTERS ?
```

```
3362  013220  001420                          BEQ     2$              ;BR IF NONE
3363  013222  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3364  013224  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3365  013226  013701  013144                  MOV     SAVER1,R1       ;DISPLAY SAVED R1
3366  013232  013705  013146                  MOV     SAVER5,R5       ;DISPLAY SAVED R5
3367  013236  004737  013274                  JSR     PC,CMPRT1       ;PRINT SAVED VALUES
3368  013242  005237  013150                  INC     ERCTR           ;INCREMENT THE ERROR COUNTER
3369  013246  005037  013144                  CLR     SAVER1          ;CLEAR SAVED REGISTER INDICATORS
3370  013252  005037  013146                  CLR     SAVER5          ;CLEAR THE OTHER ONE
3371  013256  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
3372  013260  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3373  013262  004737  013274          2$:     JSR     PC,CMPRT1       ;PRINT REMAINDER OF MESSAGE
3374  013266  005237  013150                  INC     ERCTR           ;INCREMENT THE ERROR COUNTER
3375  013272  000207                          RTS     PC              ;RETURN
3376  013274  005737  013152          CMPRT1: TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
3377  013300  001403                          BEQ     1$              ;BR IF IT HAS
3378  013302  005337  013152                  DEC      LIMIT          ;DECREMENT LIMIT COUNTER
3379  013306  001004                          BNE     2$              ;BR IF NOT AT LIMIT
3380  013310  032777  000004  165622  1$:     BIT     #SWD2,@SWR      ;PRINT ALL DATA COMPARE ERRORS ?
3381  013316  001420                          BEQ     3$              ;BR IF NOT
3382  013320  010137  001122          2$:     MOV     R1,$BDADR       ;ADDRESS OF BAD WORD
3383  013324  162737  000002  001122          SUB     #2,$BDADR       ;ADJUST ADDRESS
3384  013332  016537  177776  001124          MOV     -2(R5),$GDDAT   ;GOOD DATA
3385  013340  016137  177776  001126          MOV     -2(R1),$BDDAT   ;BAD DATA
3386  013346  112737  000013  001114          MOVB    #13,$ITEMB      ;ERROR TABLE INDEX
3387  013354  004737  012130                  JSR     PC,TYPERR       ;DISPLAY WORD THAT DIDN'T COMPARE
3388  013360  000207                  3$:     RTS     PC              ;RETURN
3389
3390                                  ;LAST LINE OF COMPARE ERROR REPORTING
3391
3392  013362  005737  013150          ENDCMP: TST     ERCTR           ;SEE IF ANY ERRORS
3393  013366  001405                          BEQ     1$              ;BR IF NONE
3394  013370  104014                          ERROR   14              ;NUMBER OF COMPARE ERRORS
3395  013372  004737  007616                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
3396  013376  062706  000002                  ADD     #2,SP           ;RETURN TO MAIN ROUTINE
3397  013402  000207                  1$:     RTS     PC              ;RETURN
3398
3399                                  ;FIND THE CORRECT PATTERN - RETURN WITH ADDRESS OF PATTERN IN R4
3400                                  ;       RETURN +2 IF PATTERN CAN'T BE FOUND
3401
3402  013404  010146                  MATCH:  MOV     R1,-(SP)        ;SAVE R1 ON THE STACK
3403  013406  012704  000044                  MOV     #44,R4          ;PATTERN TABLE INDEX
3404  013412  011601                  1$:     MOV     (SP),R1         ;RELOAD R1
3405  013414  162704  000002                  SUB     #2,R4           ;DECREMENT INDEX
3406  013420  016405  025764                  MOV     STNDAT(R4),R5   ;ADDRESS OF PATTERN ADDRESS
3407  013424  001411                          BEQ     3$              ;BR IF ALL PATTERNS CHECKED AND NO MATCH
3408                                                                  ;FOUND
3409  013426  012703  000004                  MOV     #4,R3           ;NUMBER OF LOCATIONS TO CHECK
3410  013432  022125                  2$:     CMP     (R1)+,(R5)+     ;COMPARE THE BUFFER AGAINST THE PATTERN
3411  013434  001366                          BNE     1$              ;BR IF NOT EQUAL, TRY NEXT PATTERN
3412  013436  005303                          DEC     R3              ;FINISHED CHECKING?
3413  013440  001374                          BNE     2$              ;BR IF NOT FINISHED
3414  013442  062704  025764                  ADD     #STNDAT,R4      ;MAKE PATTERN ADDRESS ABSOLUTE
3415  013446  000403                          BR      4$              ;EXIT
3416  013450  062766  000002  000002  3$:     ADD     #2,2(SP)        ;INCREMENT RETURN ADDRESS
3417  013456  012601                  4$:     MOV     (SP)+,R1        ;RESTORE R1
```

```
                                        RTS     PC              ;RETURN
                                ;ORDER TERMINATED WITH AN ERROR - PROCESS THE ERROR
013462  032760  000200  000016  ERPROC: BIT     #BIT07,STATUS(R0) ;DONE BIT SET ?
013470  001402                  BEQ     ERPRC1          ;BR IF ORDER DIDN'T COMPLETE NORMALLY
013472  000137  013662          JMP     DONE            ;PROCESS ERROR WITH 'DONE' BIT SET
                                ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE NOT' BITS
013476  032760  000400  000016  ERPRC1: BIT     #BIT08,STATUS(R0)  ;WAS A 'RESET' ISSUED ?
013504  001405                  BEQ     1$              ;BR IF NOT
013506  012777  000100  165430          MOV     #BIT06,@STKS    ;RESTORE TTY KEYBOARD INTERRUPT
013514  004737  006762                  JSR     PC,CKCLK        ;START THE CLOCK
013520  032760  040000  000016  1$:     BIT     #BIT14,STATUS(R0)  ;UNIT WENT OFFLINE ?
013526  001022                  BNE     OFLIN           ;BR IF IT DID
013530  032760  020000  000016          BIT     #BIT13,STATUS(R0)  ;BAD COMMAND CODE USED ?
013536  001023                  BNE     BADOP           ;BR IF SO
013540  032760  010000  000016          BIT     #BIT12,STATUS(R0)  ;SEE IF DRIVE WAS UNSAFE
013546  001021                  BNE     UNSAF           ;BR IF IT WENT UNSAFE
013550  032760  001000  000016          BIT     #BIT09,STATUS(R0)  ;TIMEOUT DURING POSITIONING ?
013556  001022                  BNE     TIMPOS          ;BR IF YES
013560  032760  000400  000016          BIT     #BIT8,STATUS(R0)   ;TIMEOUT DURING I/O
013566  001023                  BNE     TIMIO           ;BR IF YES
013570  104043                  ERROR   43              ;INVALID STATUS CODE
013572  000207                  RTS     PC              ;RETURN
                                ;DRIVE WENT OFFLINE
013574  104021          OFLIN:  ERROR   21              ;DRIVE WENT OFFLINE
013576  004737  007616          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
013602  000137  011226          JMP     DROP            ;DEASSIGN THE DRIVE
                                ;PROGRAM TRIED TO GIVE THE DRIVE A BAD COMMAND
013606  104002          BADOP:  ERROR   2               ;INVALID COMMAND ATTEMPTED
013610  000207                  RTS     PC              ;RETURN
                                ;DRIVE IS UNSAFE
013612  104022          UNSAF:  ERROR   22              ;DRIVE UNSAFE
013614  004737  007616          JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
013620  000137  011226          JMP     DROP            ;DEASSIGN THE DRIVE
                                ;DRIVE TIMED OUT DURING POSITIONING
013624  104017          TIMPOS: ERROR   17              ;DRIVE TIMED OUT DURING POSITIONING
013626  004737  007616          JSR     PC,INCTOT       ;INCREMENT THE 'TOTAL' ERROR COUNT
013632  000137  011226          JMP     DROP            ;DEASSIGN THE DRIVE
                                ;DRIVE TIMED OUT DURING A DATA TRANSFER
013636  104020          TIMIO:  ERROR   20              ;DRIVE TIMED OUT DURING I/O
013640  004737  007616          JSR     PC,INCTOT       ;INCREMENT THE 'TOTAL' ERROR COUNT
013644  012777  000100  165272          MOV     #BIT06,@STKS    ;RESTORE THE TTY KEYBOARD INTERRUPT ENABLE
013652  004737  006762                  JSR     PC,CKCLK        ;RESTART THE CLOCK
```

```
3474  013656  000137  011226                    JMP     DROP            ;DEASSIGN THE DRIVE
3475
3476                                    ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE' BITS SET
3477
3478  013662  022760  000002  000070  DONE:     CMP     #EOP,$RPER(R0)  ;END OF PACK ?
3479  013670  001475                            BEQ     1$              ;BR IF EOP, OPERATION OVERFLOWED
3480  013672  022760  000010  000070            CMP     #WCE,$RPER(R0)  ;WRITE CHECK ERROR ONLY ?
3481  013700  001002                            BNE     .+6             ;BR IF NOT
3482  013702  000137  014066                    JMP     WCEER           ;REPORT THE WRITE CHECK ERROR
3483  013706  032760  100000  000070            BIT     #WPV,$RPER(R0)  ;WRITE PROTECTION VIOLATION ?
3484  013714  001402                            BEQ     .+6             ;BR IF NOT
3485  013716  000137  014076                    JMP     WPVER           ;REPORT THE WRITE PROTECTION VIOLATION
3486  013722  032760  040000  000070            BIT     #FUV,$RPER(R0)  ;FILE UNSAFE ?
3487  013730  001402                            BEQ     .+6             ;BR IF NOT
3488  013732  000137  013612                    JMP     UNSAF           ;REPORT THE UNSAFE
3489  013736  032760  004000  000066            BIT     #SUSI,$RPDS(R0) ;SEEK INCOMPLETE ?
3490  013744  001402                            BEQ     .+6             ;BR IF NOT
3491  013746  000137  014424                    JMP     SKIER           ;PROCESS IT
3492  013752  032760  034000  000070            BIT     #NXC!NXT!NXS,$RPER(R0)  ;INVALID DISK ADDRESS ?
3493  013760  001402                            BEQ     .+6             ;BR IF NOT
3494  013762  000137  014106                    JMP     BDADR           ;REPORT INVALID DISK ADDRESS
3495  013766  032760  012000  000070            BIT     #PROG!MODE,$RPER(R0)  ;CONTROLLER ERROR ?
3496  013774  001402                            BEQ     .+6             ;BR IF NOT
3497  013776  000137  014116                    JMP     CNTRLR          ;REPORT THE CONTROLLER ERROR
3498  014002  032760  000024  000070            BIT     #TIMEE!NXME,$RPER(R0)  ;INTERFACE (UNIBUS) ERROR ?
3499  014010  001402                            BEQ     .+6             ;BR IF NOT
3500  014012  000137  014126                    JMP     INTRFA          ;REPORT THE ERROR
3501  014016  032760  001000  000070            BIT     #FMTE,$RPER(R0) ;FORMAT ERROR ?
3502  014024  001402                            BEQ     .+6             ;BR IF NOT
3503  014026  000137  014136                    JMP     FMTERR          ;PROCESS THE FORMAT ERROR
3504  014032  032760  010000  000066            BIT     #HNF,$RPDS(R0)  ;HEADER NOT FOUND ?
3505  014040  001402                            BEQ     .+6             ;BR IF NOT SET
3506  014042  000137  014460                    JMP     CKHNF           ;PROCESS THE ERROR
3507  014046  032760  000340  000070            BIT     #LPE!WPE!CSME,$RPER(R0)  ;DATA ERROR ?
3508  014054  001402                            BEQ     .+6             ;BR IF NOT
3509  014056  000137  014622                    JMP     DATER           ;REPORT THE DATA ERROR
3510  014062  104043                            ERROR   43              ;ERROR REPORT BUT NO BITS SET
3511  014064  000207                    1$:     RTS     PC              ;RETURN
3512
3513                                    ;REPORT A WRITE CHECK ERROR (DATA CHECK BITS NOT SET)
3514
3515  014066  104032                    WCEER:  ERROR   32              ;WRITE CHECK (DATA CHECK BITS NOT SET)
3516  014070  004737  007616                    JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3517  014074  000207                            RTS     PC              ;RETURN
3518
3519                                    ;REPORT WRITE PROTECTION VIOLATION
3520
3521  014076  104044                    WPVER:  ERROR   44              ;WRITE PROTECTION VIOLATION
3522  014100  004737  007616                    JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3523  014104  000207                            RTS     PC              ;RETURN
3524
3525                                    ;REPORT INVALID DISK ADDRESS ('NXC', 'NXT', 'NXS')
3526
3527  014106  104026                    BDADR:  ERROR   26              ;INVALID DISK ADDRESS
3528  014110  004737  007616                    JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3529  014114  000207                            RTS     PC              ;RETURN
```

```
                                        ;REPORT CONTROLLER ERROR ('PROG' & 'MODE')

014116  104025                  CNTRLR: ERROR   25              ;REPORT CONTROLLER ERROR
014120  004737  007616                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
014124  000207                          RTS     PC              ;RETURN

                                        ;REPORT UNIBUS INTERFACE ERROR ('TIMEE' & 'NXME')

014126  104024                  INTRFA: ERROR   24              ;REPORT UNIBUS ERROR
014130  004737  007616                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
014134  000207                          RTS     PC              ;RETURN

                                        ;REPORT FORMAT ERROR

014136  122760  000007  000002  FMTERR: CMPB    #WRTCHK,$COMND(R0)  ;WRITE CHECK OPERATION ?
014144  001404                          BEQ     1$              ;BR IF WRITE CHECK
014146  132760  000004  000002          BITB    #BIT02,$COMND(R0)   ;WRITE COMMAND ?
014154  001010                          BNE     2$              ;BR IF NOT
014156  016046  000076          1$:     MOV     $RPBA(R0),-(SP) ;PUT PRESENT BUFFER ADDRESS ON THE STACK
014162  166016  000006                  SUB     $BUF(R0),(SP)   ;SUBTRACT STARTING ADDRESS
014166  022726  000200                  CMP     #128.,(SP)+     ;SILO FILLED OR HAS XFER TO DISK STARTED ?
014172  103074                          BHIS    6$              ;BR IF NOT
014174  000404                          BR      3$              ;
014176  026060  000006  000076  2$:     CMP     $BUF(R0),$RPBA(R0)  ;READ TRANSFER STARTED ?
014204  001467                          BEQ     6$              ;BR IF NOT
014206  104045                  3$:     ERROR   45              ;FORMAT ERROR DURING I/O
014210  162706  000004                  SUB     #4,SP           ;MAKE ROOM ON THE STACK
014214  004737  006666                  JSR     PC,READDR       ;ADJUST SECTOR/TRACK ADDRESSES
014220  112660  000013                  MOVB    (SP)+,$TRK(R0)  ;MOVE THE TRACK ADDRESS
014224  112660  000012                  MOVB    (SP)+,$SEC(R0)  ;MOVE THE SECTOR ADDRESS
014230  016060  000106  000010          MOV     $SUCA(R0),$CYL(R0)  ;USE PRESENT CYLINDER ADDRESS
014236  022760  000400  000020          CMP     #256.,$WRDL(R0) ;WAS INITIAL TRANSFER TO BE GREATER
                                                                ;THAN 1 SECTOR ?
014244  103015                          BHIS    4$              ;BR IF NOT
014246  004737  005056                  JSR     PC,RELBUF       ;RELEASE PRESENT BUFFER
014252  012760  000400  000020          MOV     #256.,$WRDL(R0) ;SETUP FOR THE FAILING SECTOR
014260  012760  177400  000004          MOV     #-256.,$WRDM(R0)  ;WORD COUNT (2'S COMPLEMENT)
014266  005046                          CLR     -(SP)           ;CLEAR THE STACK
014270  004737  004720                  JSR     PC,GETBUF       ;GET NEW BUFFER
014274  012660  000006                  MOV     (SP)+,$BUF(R0)  ;LOAD BUFFER ADDRESS
014300  012737  001001  015572  4$:     MOV     #FMTE!DSKERR,MASK       ;ERROR MASK FOR RETRY
014306  112737  000012  015571          MOVB    #10.,$RETRY+1   ;RETRY LIMIT
014314  004737  007616                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
014320  004737  015472                  JSR     PC,RETRY        ;RETRY THE OPERATION
014324  000411                          BR      5$              ;RETRY GOOD
014326  004737  015350                  JSR     PC,RECALT       ;RESTORE (PART OF STANDARD RECOVERY)
014332  004737  015472                  JSR     PC,RETRY        ;RETRY AGAIN
014336  000404                          BR      5$              ;RETRY GOOD
014340  104042                          ERROR   42              ;UNSUCESSFUL RETRY
014342  004737  007544                  JSR     PC,INCHRD       ;INCREMENT 'HARD' ERROR COUNT
014346  000425                          BR      8$              ;EXIT
014350  104041                  5$:     ERROR   41              ;SUCESSFUL RETRY
014352  004737  007526                  JSR     PC,INCSOF       ;INCREMENT 'SOFT' ERROR COUNT
014356  004737  012534                  JSR     PC,CMPAR        ;COMPARE THE BUFFER
014362  000417                          BR      8$              ;EXIT
```

```
3586  014364  104035                  6$:     ERROR   35              ;FORMAT ERROR SEARCHING FOR INITIAL SECTOR
3587  014366  004737  007616                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3588  014372  012737  001001  015572          MOV     #FMTE!DSKERR,MASK  ;ERROR MASK FOR RETRY
3589  014400  112737  000001  015571          MOVB    #1,$RETRY+1     ;RETRY COUNT
3590  014406  004737  015472                  JSR     PC,RETRY        ;RETRY THE ORDER
3591  014412  000402                          BR      7$              ;GOOD RETRY
3592  014414  104042                          ERROR   42              ;UNSUCESSFUL RETRY
3593  014416  000401                          BR      8$              ;EXIT
3594  014420  104041                  7$:     ERROR   41              ;SUCESSFUL RETRY
3595  014422  000207                  8$:     RTS     PC              ;RETURN
3596
3597                                  ;REPORT SEEK INCOMPLETE
3598
3599  014424  016037  000032  001124  SKIER:  MOV     $PREVA+2(R0),$GDDAT     ;PREVIOUS CYLINDER ADDRESS
3600  014432  016037  000010  001126          MOV     $CYL(R0),$BDDAT ;DESTINATION CYLINDER
3601  014440  104023                          ERROR   23              ;SEEK INCOMPLETE
3602  014442  004737  007616                  JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3603  014446  004737  007562                  JSR     PC,INCSKI       ;INCREMENT 'SUSI' ERROR COUNT
3604  014452  004737  015350                  JSR     PC,RECALT       ;RESTORE
3605  014456  000207                          RTS     PC              ;RETURN
3606
3607                                  ;REPORT HEADER NOT FOUND ERROR OR POSITIONING ERROR
3608
3609  014460  116046  000102          CKHNF:  MOVB    $RPDA(R0),-(SP) ;SECTOR ADDRESS ON THE STACK
3610  014464  042716  000360                  BIC     #360,(SP)       ;CLEAR THE 'SOT' BITS
3611  014470  116046  000103                  MOVB    $RPDA+1(R0),-(SP)  ;TRACK ADDRESS
3612  014474  004737  015402                  JSR     PC,READHD       ;READ THE HEADER FROM THE INITIAL SECTOR
3613  014500  032737  100000  025436          BIT     #ERR,GENREG+RPER  ;ERROR DURING OPERATION ?
3614  014506  001041                          BNE     1$              ;BR IF ERROR
3615  014510  013737  033640  001126          MOV     CYLDER+2,$BDDAT ;CYLINDER FROM HEADER
3616  014516  006237  001126                  ASR     $BDDAT          ;SHIFT CYLINDER ADDRESS
3617  014522  006237  001126                  ASR     $BDDAT          ;SHIFT CYLINDER ADDRESS
3618  014526  006237  001126                  ASR     $BDDAT          ;SHIFT CYLINDER ADDRESS
3619  014532  006237  001126                  ASR     $BDDAT          ;SHIFT CYLINDER ADDRESS
3620  014536  006237  001126                  ASR     $BDDAT          ;SHIFT CYLINDER ADDRESS
3621  014542  006237  001126                  ASR     $BDDAT          ;SHIFT CYLINDER ADDRESS
3622  014546  026037  000106  001126          CMP     $SUCA(R0),$BDDAT  ;AT CORRECT CYLINDER ?
3623  014554  001416                          BEQ     1$              ;BR IF AT CORRECT CYLINDER
3624  014556  016037  000032  001124          MOV     $PREVA+2(R0),$GDDAT  ;PREVIOUS CYLINDER ADDRESS
3625  014564  104036                          ERROR   36              ;POSITIONING ERROR
3626  014566  004737  007600                  JSR     PC,INCMIS       ;INCREMENT POSITIONING ERROR COUNT
3627  014572  004737  015350                  JSR     PC,RECALT       ;RESTORE THE DRIVE
3628  014576  016037  000106  000032          MOV     $SUCA(R0),$PREVA+2  ;SAVE PREVIOUS CYLINDER
3629  014604  005060  000010                  CLR     $CYL(R0)        ;RESET CURRENT CYLINDER
3630  014610  000401                          BR      2$              ;CLEAN UP, THEN EXIT
3631  014612  104034                  1$:     ERROR   34              ;HEADER NOT FOUND ERROR
3632  014614  004737  007616          2$:     JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3633  014620  000207                          RTS     PC              ;RETURN
3634
3635                                  ;REPORT DATA ERROR ('LPE', 'WPE', & 'CSME')
3636
3637  014622  032760  000010  000070  DATER:  BIT     #WCE,$RPER(R0)  ;WRITE CHECK ERROR ALSO ?
3638  014630  001402                          BEQ     1$              ;BR IF NOT
3639  014632  104033                          ERROR   33              ;REPORT WRITE CHECK WITH DATA CHECK
3640  014634  000403                          BR      2$              ;CONTINUE
3641  014636  104027                  1$:     ERROR   27              ;DATA ERROR
```

```
3642  014640  004737  015144              JSR     PC,PRTBAD       ;TYPE THE BAD SECTOR
3643  014644  004737  007616      2$:     JSR     PC,INCTOT       ;INCREMENT 'TOTAL' ERROR COUNT
3644  014650  162706  000004              SUB     #4,SP           ;MAKE ROOM ON THE STACK
3645  014654  004737  006666              JSR     PC,READDR       ;ADJUST SECTOR/TRACK ADDRESSES
3646  014660  112660  000013              MOVB    (SP)+,$TRK(R0)  ;MOVE THE TRACK ADDRESS
3647  014664  112660  000012              MOVB    (SP)+,$SEC(R0)  ;MOVE THE SECTOR ADDRESS
3648  014670  016060  000106  000010      MOV     $SUCA(R0),$CYL(R0)  ;USE PRESENT CYLINDER ADDRESS
3649  014676  022760  000400  000020      CMP     #256.,$WROL(R0) ;INITIAL TRANSFER SIZE LARGER
3650                                                              ;THAN 1 SECTOR ?
3651  014704  103023                      BHIS    3$              ;BR IF NOT
3652  014706  004737  005056              JSR     PC,RELBUF       ;RELEASE THE PRESENT BUFFER
3653  014712  012760  000400  000020      MOV     #256.,$WROL(R0)  ;SETUP FOR 1 SECTOR
3654  014720  012760  177400  000004      MOV     #-256.,$WRDM(R0)  ;WORD COUNT
3655  014726  005046                      CLR     -(SP)           ;CLEAR THE STACK
3656  014730  004737  004720              JSR     PC,GETBUF       ;GET NEW BUFFER
3657  014734  012660  000006              MOV     (SP)+,$BUF(R0)  ;LOAD BUFFER ADDRESS
3658  014740  122760  000007  000002      CMPB    #WRTCHK,$COMND(R0)  ;IS THE COMMAND A WRITE CHECK ?
3659  014746  001002                      BNE     3$              ;BR IF NOT
3660  014750  004737  005304              JSR     PC,FILBUF       ;REFILL THE BUFFER
3661  014754  012737  000350  015572  3$: MOV     #WCE!LPE!WPE!CSME,MASK  ;RETRY MASK
3662  014762  112737  000012  015571      MOVB    #10.,$RETRY+1   ;RETRY LIMIT
3663  014770  004737  015472              JSR     PC,RETRY        ;RETRY THE SECTOR
3664  014774  000413                      BR      4$              ;RETRY GOOD
3665  014776  004737  015350              JSR     PC,RECALT       ;RESTORE THE DRIVE
3666  015002  004737  015472              JSR     PC,RETRY        ;RETRY 10 MORE TIMES
3667  015006  000406                      BR      4$              ;RETRY GOOD
3668  015010  104031                      ERROR   31              ;HARD ERROR
3669  015012  004737  007544              JSR     PC,INCHRD       ;INCREMENT 'HARD' ERROR COUNT
3670  015016  004737  015144              JSR     PC,PRTBAD       ;PRINT BAD SECTOR ?
3671  015022  000405                      BR      5$              ;EXIT
3672  015024  104030              4$:     ERROR   30              ;SOFT ERROR
3673  015026  004737  007526              JSR     PC,INCSOF       ;INCREMENT 'SOFT' ERROR COUNT
3674  015032  004737  012534              JSR     PC,CMPAR        ;COMPARE THE BUFFER
3675  015036  000207              5$:     RTS     PC              ;RETURN
3676
3677
3678                                      ;REPORT AN 'UNKNOWN' DATA PATTERN
3679
3680  015040  010346              NOMTCH: MOV     R3,-(SP)        ;SAVE R3
3681  015042  112737  000010  001114      MOVB    #10,$ITEMB      ;ERROR TABLE INDEX
3682  015050  004737  012130              JSR     PC,TYPERR       ;CAN'T MATCH PATTERN WITH DATA READ
3683  015054  012703  000004              MOV     #4,R3           ;WORD COUNTER
3684  015060  010137  001122      1$:     MOV     R1,$BDADR       ;ADDRESS OF WORD
3685  015064  012137  001126              MOV     (R1)+,$BDDAT    ;WORD
3686  015070  112737  000011  001114      MOVB    #11,$ITEMB      ;ERROR TABLE INDEX
3687  015076  004737  012130              JSR     PC,TYPERR       ;DISPLAY THE PATTERN 'SEED'
3688  015102  005303                      DEC     R3              ;DECREMENT WORD COUNTER
3689  015104  001365                      BNE     1$              ;BR IF NOT DONE
3690  015106  062701  000770              ADD     #770,R1         ;INCREMENT BUFFER POINTER
3691  015112  060237  013150              ADD     R2,ERCTR        ;ADD SECTOR (OR LESS) COUNT TO THE
3692                                                              ;COMPARSION ERROR COUNT
3693  015116  005002                      CLR     R2              ;CLEAR 'WORDS TO COMPARE' COUNT IN R2
3694  015120  112737  177777  013143      MOVB    #-1,FRSTER      ;SET 'NOT FIRST ERROR' INDICATOR
3695  015126  012603                      MOV     (SP)+,R3        ;RESTORE R3
3696  015130  032777  100000  164002      BIT     #SW15,@SWR      ;HALT ON ERROR ?
3697  015136  001401                      BEQ     2$              ;BR IF NOT
```

```
                                                HALT                            ;ERROR HALT
                                        2$:     RTS     PC                      ;RETURN

                                        ;ROUTINE TO DISPLAY THE SECTOR WHICH GAVE THE HARD ERROR

                                PRTBAD: BIT     #SW3,@SWR               ;PRINT THE BAD SECTOR ?
                                        BEQ     7$                      ;BR IF NOT
                                        BIT     #SW13,@SWR              ;INHIBIT ERROR TYPEOUTS
                                        BNE     7$                      ;BR IF INHIBIT
                                        CMPB    #WRTCHK,$COMND(R0)      ;PRESENT COMMAND A WRITE CHECK ?
                                        BEQ     7$                      ;BR IF IT IS
                                        BIT     #BIT02,$COMND(R0)       ;PRESENT COMMAND A WRITE ?
                                        BEQ     7$                      ;BR IF IT IS
                                        MOV     $RPBA(R0),R1            ;PUT THE END ADDRESS INTO R1
                                        MOV     $WRDL(R0),-(SP)         ;FIND THE BEGINNING OF THE SECTOR
                                        ADD     $RPWC(R0),(SP)          ;SUBTRACT THE WORDS NOT TRANSFERED
                                        CLR     -(SP)                   ;MAKE THE UPPER DIVIDEND 0
                                        MOV     #256.,-(SP)     ;DIVIDE THE WORDS TRANSFERED BY THE SECTOR SIZE
                                        JSR     PC,$DIV                 ;DIVIDE
                                        TST     (SP)                    ;REMAINDER = 0 ?
                                        BEQ     1$                      ;BR IF IT IS - COMPLETE SECTOR TRANSFERED
                                        ASL     (SP)                    ;CONVERT THE RESIDUAL SECTOR SIZE INTO BYTE COUNT
                                        SUB     (SP),R1                 ;SUBTRACT IT FROM THE END ADDRESS
                                        BR      2$                      ;FINISH THE SIZING
                                1$:     SUB     #1000,R1                ;SUBTRACT FULL SECTOR SIZE FROM END ADDR
                                2$:     ADD     #4,SP                   ;RESTORE THE STACK POINTER
                                        TYPE    ,LIN11H                 ;PRINT THE HEADER
                                3$:     BIT     #SW03,@SWR              ;PRINT ANY MORE OF THE SECTOR
                                        BEQ     5$                      ;BR IF NOT
                                        MOV     #7,R2                   ;R2 CONTAINS THE WORDS/LINE COUNT
                                        MOV     R1,-(SP)                ;PUT THE ADDRESS ON THE STACK
                                        TYPOC                           ;PRINT IT
                                4$:     CMP     R1,$RPBA(R0)            ;PRINTED ALL THE SECTOR ?
                                        BEQ     5$                      ;BR IF ALL PRINTED
                                        TYPE    ,SPACE2                 ;SPACES
                                        MOV     (R1)+,-(SP)             ;PUT THE DATA ON THE STACK
                                        TYPOC                           ;PRINT IT
                                        DEC     R2                      ;DECREMENT THE HORIZONTAL COUNT
                                        BNE     4$                      ;BR IF NOT AT THE END OF THE LINE
                                        TYPE    ,$CRLF                  ;CR-LF
                                        BR      3$                      ;RESTORE THE WORDS/LINE COUNT
                                5$:     TYPE    ,$CRLF                  ;PRINT WHAT REMAINS IN THE BUFFER
                                6$:     BIT     #SW15,@SWR              ;HALT ON ERROR ?
                                        BEQ     7$                      ;BR IF NOT
                                        HALT                            ;HALT
                                7$:     RTS     PC                      ;RETURN

                                        ;ROUTINE TO DO A HOME SEEK - DRIVE SELECTED IN R0
                                                ;ENTER AT 'RECAL0' IF UNIT NUMBER ALREADY LOADED
                                                ;INTO 'GENDPB'

                                RECALT: MOVB    (R0),GENDPB             ;MOVE THE UNIT # TO THE GENERAL DPB
                                RECAL0: MOVB    #HOMSEK,GENDPB+$COMND   ;HOME SEEK COMMAND
                                        JSR     R0,RP11                 ;DRIVER ENTRANCE
                                        GENDPB                          ;DPB ADDRESS FOR ORDER
                                        HALT                            ;DRIVER DIDN'T ACCEPT THE ORDER
```

```
3754  015372  005737  025432          1$:     TST     GENDPB+$STATUS          ;SEE IF FINISHED
3755  015376  001775                          BEQ     1$              ;BR IF NOT FINISHED
3756  015400  000207                          RTS     PC              ;RETURN
3757
3758                                  ;UTILITY READ HEADER ROUTINE
3759                                  ;       ENTER WITH TRACK AND SECTOR ADDRS ON THE STACK
3760
3761  015402  116637  000002  025427  READHD: MOVB    2(SP),GENDPB+$TRK ;TRACK ADDRESS
3762  015410  116637  000004  025426          MOVB    4(SP),GENDPB+$SEC ;SECTOR ADDRESS
3763  015416  111037  025414                  MOVB    (R0),GENDPB     ;DRIVE NUMBER
3764  015422  016037  000106  025424          MOV     $SUCA(R0),GENDPB+$CYL ;CYLINDER ADDRESS
3765  015430  012737  014000  025416          MOV     #MODE!HDR,GENDPB+$COMND  ;MODE AND HEADER BITS
3766  015436  112737  000017  025416          MOVB    #READ,GENDPB+$COMND ;READ COMMAND
3767  015444  004037  021566                  JSR     R0,RP11         ;DRIVER ENTRANCE
3768  015450  025414                          GENDPB                  ;DPB ADDRESS FOR ORDER
3769  015452  000000                          HALT                    ;DRIVER DIDN'T ACCEPT COMMAND
3770  015454  005737  025432          1$:     TST     GENDPB+$STATUS  ;FINISHED?
3771  015460  001775                          BEQ     1$              ;BR IF NOT
3772  015462  012666  000002                  MOV     (SP)+,2(SP)     ;ADJUST STACK FOR RETURN
3773  015466  005726                          TST     (SP)+
3774  015470  000207                          RTS     PC              ;RETURN
3775
3776                                  ;RETRY THE PRESENT OPERATION
3777                                  ;       RETURN IF RETRY SUCESSFUL
3778                                  ;       RETURN+2 IF RETRY UNSUCESSFUL
3779                                  ;       RETURN TO MAIN PROGRAM IF DIFFERENT ERROR OCCURS
3780
3781  015472  105037  015570  RETRY:  1$:     CLRB    $RETRY          ;CLEAR RETRY COUNTER
3782  015476  004737  005434          JSR     PC,GOORIV       ;RE-START ORDER
3783  015502  005760  000016          2$:     TST     STATUS(R0)      ;ORDER FINISHED?
3784  015506  001775                          BEQ     2$              ;BR IF NOT
3785  015510  100403                          BMI     3$              ;BR IF ERROR
3786  015512  105237  015570                  INCB    $RETRY          ;INCREMENT RETRY COUNT
3787  015516  000420                          BR      6$              ;GO TO EXIT, SUCESSFUL RETRY
3788  015520  032760  000200  000016  3$:     BIT     #BIT7,STATUS(R0)  ;DID ORDER TERMINATE NORMALLY ?
3789  015526  001415                          BEQ     7$              ;BR IF NOT
3790  015530  033760  015572  000070          BIT     MASK,$RPER(R0)  ;SAME ERROR?
3791  015536  001411                          BEQ     7$              ;BR IF NOT
3792  015540  105237  015570          5$:     INCB    $RETRY          ;INCREMENT RETRY COUNT
3793  015544  123737  015570  015571          CMPB    $RETRY,$RETRY+1 ;REACHED RETRY LIMIT ?
3794  015552  001361                          BNE     1$              ;BR IF NOT
3795  015554  062716  000002                  ADD     #2,(SP)         ;INCREMENT RETURN, UNSUCESSFUL RETRY
3796  015560  000207                  6$:     RTS     PC              ;RETURN
3797  015562  104037                  7$:     ERROR   37              ;DIFFERENT ERROR DURING RETRY
3798  015564  005726                          TST     (SP)+           ;ADJUST STACK POINTER FOR DIRECT RETURN
3799  015566  000207                          RTS     PC              ;RETURN
3800
3801  015570     000     000          $RETRY: .BYTE   0,0             ;RETRY LIMIT & RETRY COUNTER
3802  015572  000000                  MASK:   .WORD   0               ;ERROR MASK
3803
3804                                  ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO
3805                                  ;AN UNSIGNED DECIMAL ASCIZ NUMBER.
3806                                  ;
3807                                  ;CALL
3808                                  ;       MOV     #NUMBER,-(SP)   ;PUT THE BINARY NUMBER ON THE STACK
3809                                  ;       JSR     PC,$$B2D        ;CALL
```

```
                                ;       RETURN                  ;ADDRESS OF 1ST CHARACTER IS ON THE STACK
                                ;
                                ;NOTE:  THIS FORM OF '$SB2D' MUST BE USED, THE PROGRAM CANNOT USE THE
                                ;VERSION OF THIS ROUTINE WHICH IS ON THE SYSMAC LIBRARY, REV C OR LATER.
                                ;
015574  016637  000002  015620  $SB2D:  MOV     2(SP),1$        ;SAVE THE BINARY NUMBER
015602  012746  015620                  MOV     #1$,-(SP)       ;SET POINTER
015606  004737  020624                  JSR     PC,$DB2D        ;CALL DOUBLE LENGTH CONVERT
015612  012666  000002                  MOV     (SP)+,2(SP)     ;PICKUP THE POINTER
015616  000207                          RTS     PC              ;RETURN
015620  000000  000000          1$:     .WORD   0,0
                                ;;**************************************************************
                                .SBTTL  MACRO ROUTINES
                                .SBTTL  ERROR HANDLER ROUTINE
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO TYPERR ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
                                ;
015624                  $ERROR:
015624  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
015626  010146                          MOV     R1,-(SP)        ;SAVE R1
015630  010246                          MOV     R2,-(SP)        ;SAVE R2
015632  111037  001264                  MOVB    (R0),DRIVE      ;DRIVE NUMBER
015636  162706  000004                  SUB     #4,SP           ;MAKE ROOM ON THE STACK
015642  004737  006666                  JSR     PC,READDR       ;DECREMENT THE TRACK/SECTOR ADDRESSES
015646  112637  001270                  MOVB    (SP)+,TRK.ER    ;TRACK ADDRESS
015652  112637  001272                  MOVB    (SP)+,SEC.ER    ;SECTOR ADDRESS
015656  016037  000106  001266          MOV     $SUCA(R0),CYL.ER  ;CYLINDER ADDRESS
015664  012701  001162                  MOV     #$REGO,R1       ;START AT '$REGO'
015670  016021  000066                  MOV     $RPDS(R0),(R1)+ ;MOVE RPDS
015674  016021  000070                  MOV     $RPER(R0),(R1)+ ;MOVE RPER
015700  016021  000072                  MOV     $RPCS(R0),(R1)+ ;MOVE RPCS
015704  016021  000074                  MOV     $RPWC(R0),(R1)+ ;MOVE RPWC
015710  016021  000076                  MOV     $RPBA(R0),(R1)+ ;MOVE RPBA
015714  016021  000100                  MOV     $RPCA(R0),(R1)+ ;MOVE RPCA
015720  016021  000102                  MOV     $RPDA(R0),(R1)+ ;MOVE RPDA
015724  016021  000104                  MOV     $RPMI(R0),(R1)+ ;MOVE RPMI
015730  016021  000106                  MOV     $SUCA(R0),(R1)+ ;MOVE SUCA
015734  016021  000110                  MOV     $SILO(R0),(R1)+ ;MOVE SILO
015740  012602                          MOV     (SP)+,R2        ;RESTORE R2
015742  012601                          MOV     (SP)+,R1        ;RESTORE R1
015744  105237  001103          7$:     INCB    SERFLG          ;;SET THE ERROR FLAG
015750  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015752  013777  001102  163162          MOV     STSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
015760  032777  002000  163152          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
```

```
 3866  015766  001402                        BEQ     1$              ;;NO - SKIP
 3867  015770  104401  001206                TYPE    ,$BELL          ;;RING BELL
 3868  015774  005237  001112        1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 3869  016000  011637  001116                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 3870  016004  162737  000002  001116        SUB     #2,$ERRPC
 3871  016012  117737  163100  001114        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 3872  016020  032777  020000  163112        BIT     #BIT13,@$SWR    ;;SKIP TYPEOUT IF SET
 3873  016026  001004                        BNE     20$             ;;SKIP TYPEOUTS
 3874  016030  004737  012130                JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
 3875  016034  104401  001213                TYPE    ,$CRLF
 3876  016040                        20$:
 3877  016040  005777  163074        2$:     TST     @$SWR           ;;HALT ON ERROR
 3878  016044  100002                        BPL     3$              ;;SKIP IF CONTINUE
 3879  016046  000000                        HALT                    ;;HALT ON ERROR!
 3880  016050  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 3881  016052                        3$:
 3882  016052  000002                        RTI                     ;;RETURN
 3883
 3884                                  .SBTTL  TYPE ROUTINE
 3885
 3886                                  ;;**************************************************************
 3887                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 3888                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 3889                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 3890                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 3891                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 3892                                  ;*
 3893                                  ;*CALL:
 3894                                  ;*1) USING A TRAP INSTRUCTION
 3895                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 3896                                  ;*OR
 3897                                  ;*      TYPE
 3898                                  ;*      MESADR
 3899                                  ;*
 3900
 3901  016054  105737  001157        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 3902  016060  100002                        BPL     1$              ;;BR IF YES
 3903  016062  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
 3904  016064  000407                        BR      3$              ;;LEAVE
 3905  016066  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
 3906  016070  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 3907  016074  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 3908  016076  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 3909  016100  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 3910  016102  012600                6$:     MOV     (SP)+,R0        ;;RESTORE R0
 3911  016104  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 3912  016110  000002                        RTI                     ;;RETURN
 3913  016112  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 3914  016116  001430                        BEQ     8$
 3915  016120  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 3916  016124  001006                        BNE     5$
 3917  016126  005726                        TST     (SP)+           ;;POP  (CR)<LF> EQUIV
 3918  016130  104401                        TYPE                    ;;TYPE A CR AND LF
 3919  016132  001213                        $CRLF
 3920  016134  105037  016270                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 3921  016140  000755                        BR      2$              ;;GET NEXT CHARACTER
```

```
                                        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
                                        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
                                                BNE     2$              ;;IF NO GO GET NEXT CHAR.
                                                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                        ;;AND THE NULL CHAR.
                                        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
                                                BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
                                                JSR     PC,$TYPEC       ;;GO TYPE A NULL
                                                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
                                                BR      7$              ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

                                        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
                                        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
                                                BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
                                                BNE     9$              ;;TAB STOP
                                                TST     (SP)+           ;;POP SPACE OFF STACK
                                                BR      2$              ;;GET NEXT CHARACTER
                                        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
                                                BPL     $TYPEC
                                                MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
                                                CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
                                                BNE     1$              ;;BRANCH IF NO
                                                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
                                                BR      $TYPEX          ;;EXIT
                                        1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
                                                BEQ     $TYPEX          ;;BRANCH IF YES
                                                INCB    (PC)+           ;;COUNT THE CHARACTER
                                        $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
                                        $TYPEX: RTS     PC

                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

```
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                                        ;
016274 017646 000000                    STYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016300 116637 000001 016517             MOVB    1(SP),$OFILL            ;;LOAD ZERO FILL SWITCH
016306 112637 016521                    MOVB    (SP)+,$OMODE+1          ;;NUMBER OF DIGITS TO TYPE
016312 062716 000002                    ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
016316 000406                           BR      STYPON
016320 112737 000001 016517     STYPOC: MOVB    #1,$OFILL               ;;SET THE ZERO FILL SWITCH
016326 112737 000006 016521             MOVB    #6,$OMODE+1             ;;SET FOR SIX(6) DIGITS
016334 112737 000005 016516     STYPON: MOVB    #5,$OCNT                ;;SET THE ITERATION COUNT
016342 010346                           MOV     R3,-(SP)                ;;SAVE R3
016344 010446                           MOV     R4,-(SP)                ;;SAVE R4
016346 010546                           MOV     R5,-(SP)                ;;SAVE R5
016350 113704 016521                    MOVB    $OMODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
016354 005404                           NEG     R4
016356 062704 000006                    ADD     #6,R4                   ;;SUBTRACT IT FOR MAX. ALLOWED
016362 110437 016520                    MOVB    R4,$OMODE               ;;SAVE IT FOR USE
016366 113704 016517                    MOVB    $OFILL,R4               ;;GET THE ZERO FILL SWITCH
016372 016605 000012                    MOV     12(SP),R5               ;;PICKUP THE INPUT NUMBER
016376 005003                           CLR     R3                      ;;CLEAR THE OUTPUT WORD
016400 006105                   1$:     ROL     R5                      ;;ROTATE MSB INTO "C"
016402 000404                           BR      3$                      ;;GO DO MSB
016404 006105                   2$:     ROL     R5                      ;;FORM THIS DIGIT
016406 006105                           ROL     R5
016410 006105                           ROL     R5
016412 010503                           MOV     R5,R3
016414 006103                   3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
016416 105337 016520                    DECB    $OMODE                  ;;TYPE THIS DIGIT?
016422 100016                           BPL     7$                      ;;BR IF NO
016424 042703 177770                    BIC     #177770,R3              ;;GET RID OF JUNK
016430 001002                           BNE     4$                      ;;TEST FOR 0
016432 005704                           TST     R4                      ;;SUPPRESS THIS 0?
016434 001403                           BEQ     5$                      ;;BR IF YES
016436 005204                   4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
016440 052703 000060                    BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
016444 052703 000040                    BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
016450 110337 016514                    MOVB    R3,8$                   ;;SAVE FOR TYPING
016454 104401 016514                    TYPE    ,8$                     ;;GO TYPE THIS DIGIT
016460 105337 016516            7$:     DECB    $OCNT                   ;;COUNT BY 1
016464 003347                           BGT     2$                      ;;BR IF MORE TO DO
016466 002402                           BLT     6$                      ;;BR IF DONE
016470 005204                           INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
016472 000744                           BR      2$                      ;;GO DO THE LAST DIGIT
016474 012605                   6$:     MOV     (SP)+,R5                ;;RESTORE R5
016476 012604                           MOV     (SP)+,R4                ;;RESTORE R4
016500 012603                           MOV     (SP)+,R3                ;;RESTORE R3
016502 016666 000002 000004             MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
016510 012616                           MOV     (SP)+,(SP)
016512 000002                           RTI                             ;;RETURN
016514     000                  8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
016515     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
016516     000                  $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
016517     000                  $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
016520 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
                                
016522                          $TYPDS:
016522  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016524  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016526  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016530  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016532  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016534  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
016540  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
016544  100004                          BPL     1$              ;;BR IF INPUT IS POS.
016546  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
016550  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
016556  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
016560  012703  016736                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
016564  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
016570  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
016572  016001  016726                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
016576  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
016600  002402                          BLT     4$              ;;BR IF DONE
016602  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
016604  000774                          BR      3$
016606  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
016610  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
016612  001002                          BNE     5$              ;;FALL THROUGH IF 0
016614  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
016616  100407                          BMI     7$              ;;BR IF YES
016620  106316                  5$:     ASLB    (SP)            ;;MSD?
016622  103003                          BCC     6$              ;;BR IF NO
016624  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
016632  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
016636  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
016642  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
016644  005720                          TST     (R0)+           ;;JUST INCREMENTING
016646  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
016652  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
016654  003002                          BGT     8$              ;;GO TO EXIT
016656  010502                          MOV     R5,R2           ;;GET THE LSD
016660  000764                          BR      6$              ;;GO CHANGE TO ASCII
016662  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
016664  100003                          BPL     9$              ;;BR IF NO
016666  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
016674  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
016676  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
016700  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
016702  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
016704  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
```

```
 4090  016706  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
 4091  016710  104401  016736              TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4092  016714  016666  000002  000004      MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4093  016722  012616                      MOV     (SP)+,(SP)
 4094  016724  000002                      RTI                     ;;RETURN TO USER
 4095  016726  023420              $DTBL:  10000.
 4096  016730  001750                      1000.
 4097  016732  000144                      100.
 4098  016734  000012                      10.
 4099  016736  000004              $DBLK:  .BLKW   4
 4100
 4101                              .SBTTL  TTY INPUT ROUTINE
 4102
 4103                              ;;******************************************************************
 4104                              .ENABL  LSB
 4105  016746  000000              $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
 4106  016750  000000              $TKQIN: .WORD   0               ;;INPUT POINTER
 4107  016752  000000              $TKQOUT: .WORD  0               ;;OUTPUT POINTER
 4108  016754  000001              $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
 4109          016755              $TKQEND=.
 4110          016756              .EVEN
 4111
 4112                              ;*TK INITIALIZE ROUTINE
 4113                              ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 4114                              ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 4115                              ;
 4116                              ;*CALL:
 4117                              ;*      JSR     PC,$TKINT
 4118                              ;*      RETURN
 4119                              ;
 4120  016756  005037  016746      $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 4121  016762  012737  016754  016750      MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 4122  016770  013737  016750  016752      MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
 4123  016776  012737  017026  000060      MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 4124  017004  012737  000200  000062      MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
 4125  017012  005777  162130              TST     @$TKB           ;;CLEAR DONE FLAG
 4126  017016  012777  000100  162120      MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 4127  017024  000207                      RTS     PC              ;;RETURN TO CALLER
 4128
 4129                              ;*TK SERVICE ROUTINE
 4130                              ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 4131                              ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 4132                              ;*IT IN THE QUEUE.
 4133                              ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
 4134                              ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (@CNTRLC)
 4135                              ;
 4136  017026  117746  162114      $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 4137  017032  042716  177600              BIC     #^C177,(SP)     ;;STRIP THE JUNK
 4138  017036  021627  000003              CMP     (SP),#3         ;;IS IT A CONTROL C?
 4139  017042  001007                      BNE     1$              ;;BRANCH IF NO
 4140  017044  104401  020144              TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
 4141  017050  004737  016756              JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 4142  017054  005726                      TST     (SP)+           ;;CLEAN UP STACK
 4143  017056  000177  162160              JMP     @CNTRLC         ;;CONTROL C RESTART
 4144  017062  021627  000007      1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 4145  017066  001004                      BNE     2$              ;;BRANCH IF NO
```

```
 4146  017070  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 4147  017076  001500                          BEQ     6$              ;;GO TO SWR CHANGE
 4148
 4149  017100                          2$:
 4150  017100  022737  000001  016746          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
 4151  017106  001004                          BNE     3$              ;;BRANCH IF NO
 4152  017110  104401  001206                  TYPE    $BELL           ;;RING THE TTY BELL
 4153  017114  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 4154  017116  000451                          BR      5$              ;;EXIT
 4155  017120  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 4156  017124  001021                          BNE     32$             ;;BRANCH IF NO
 4157  017126  005077  162012                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 4158  017132  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
 4159  017134  105777  162004          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 4160  017140  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
 4161  017142  117746  162000                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 4162  017146  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4163  017152  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4164  017156  001366                          BNE     31$             ;;BRANCH IF NO
 4165  017160  012777  000100  161756          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 4166  017166  000002                          RTI                     ;;RETURN
 4167  017170  005237  016746          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
 4168  017174  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
 4169  017200  002405                          BLT     4$              ;;BRANCH IF YES
 4170  017202  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
 4171  017206  003002                          BGT     4$              ;;BRANCH IF YES
 4172  017210  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
 4173  017214  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
 4174  017220  005237  016750                  INC     $TKQIN          ;;UPDATE THE POINTER
 4175  017224  023727  016750  016755          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
 4176  017232  001003                          BNE     5$              ;;BRANCH IF NO
 4177  017234  012737  016754  016750          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
 4178  017242  000002                  5$:     RTI                     ;;RETURN
 4179
 4180                                  ;;**************************************************************
 4181                                  ;#SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 4182                                  ;#ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
 4183                                  ;#SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 4184                                  ;#CALL WHEN OPERATING IN TTY INTERRUPT MODE.
 4185  017244  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
 4186  017252  001124                          BNE     15$             ;;EXIT IF NOT
 4187  017254  105777  161664                  TSTB    @$TKS           ;;IS A CHAR WAITING?
 4188  017260  100121                          BPL     15$             ;;IF NOT, EXIT
 4189  017262  117746  161660                  MOVB    @$TKB,-(SP)     ;;YES
 4190  017266  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
 4191  017272  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
 4192  017276  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
 4193                                                                  ;;AND EXIT
 4194
 4195                                  ;;**************************************************************
 4196                                  ;#CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
 4197                                  ;#ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
 4198                                  ;#CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
 4199  017300  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 4200  017306  001674                          BEQ     2$              ;;BRANCH IF YES
 4201  017310  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
```

```
4202  017312  004737  016756          JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
4203  017316  005077  161622          CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
4204  017322  112737  000001  001135  MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
4205
4206  017330  104401  020156          TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (↑G)
4207  017334  104401  020163  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
4208  017340  013746  000176          MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
4209  017344  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4210  017346  104401  020174          TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
4211  017352  005046          19$:    CLR     -(SP)           ;;CLEAR COUNTER
4212  017354  005046                  CLR     -(SP)           ;;THE NEW SWR
4213  017356  105777  161562  7$:     TSTB    @$TKS           ;;CHAR THERE?
4214  017362  100375                  BPL     7$              ;;IF NOT TRY AGAIN
4215
4216  017364  117746  161556          MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
4217  017370  042716  177600          BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
4218
4219  017374  021627  000003          CMP     (SP),#3         ;;IS IT A CONTROL-C?
4220  017400  001015                  BNE     9$              ;;BRANCH IF NOT
4221  017402  104401  020144          TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (↑C)
4222  017406  062706  000006          ADD     #6,SP           ;;CLEAN UP STACK
4223  017412  123727  001135  000001  CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
4224  017420  001003                  BNE     8$              ;;BRANCH IF NO
4225  017422  012777  000100  161514  MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
4226  017430  000177  161606  8$:     JMP     @CNTRLC         ;;CONTROL-C RESTART
4227
4228
4229  017434  021627  000025  9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
4230  017440  001005                  BNE     10$             ;;BRANCH IF NOT
4231  017442  104401  020151          TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
4232  017446  062706  000006  20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
4233  017452  000737                  BR      19$             ;;LET'S TRY IT AGAIN
4234
4235
4236  017454  021627  000015  10$:    CMP     (SP),#15        ;;IS IT A <CR>?
4237  017460  001022                  BNE     16$             ;;BRANCH IF NO
4238  017462  005766  000004          TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
4239  017466  001403                  BEQ     11$             ;;BRANCH IF YES
4240  017470  016677  000002  161442  MOV     2(SP),@SWR      ;;SAVE NEW SWR
4241  017476  062706  000006  11$:    ADD     #6,SP           ;;CLEAR UP STACK
4242  017502  104401  001213  14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
4243  017506  123727  001135  000001  CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
4244  017514  001003                  BNE     15$             ;;BRANCH IF NOT
4245  017516  012777  000100  161420  MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4246  017524  000002          15$:    RTI                     ;;RETURN
4247  017526  004737  016224  16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
4248  017532  021627  000060          CMP     (SP),#60        ;;CHAR < 0?
4249  017536  002420                  BLT     18$             ;;BRANCH IF YES
4250  017540  021627  000067          CMP     (SP),#67        ;;CHAR > 7?
4251  017544  003015                  BGT     18$             ;;BRANCH IF YES
4252  017546  042726  000060          BIC     #60,(SP)+       ;;STRIP-OFF ASCII
4253  017552  005766  000002          TST     2(SP)           ;;IS THIS THE FIRST CHAR
4254  017556  001403                  BEQ     17$             ;;BRANCH IF YES
4255  017560  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
4256  017562  006316                  ASL     (SP)            ;;   CHAR OVER TO MAKE
4257  017564  006316                  ASL     (SP)            ;;   ROOM FOR NEW ONE.
```

```
                                                ;;*****************************************************************
                                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*CALL:
                                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
                                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                                ;
017606  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
017610  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
017616  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
017622  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
017624  012746  017632                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
017630  000002                          RTI                     ;;POP NEW PC AND PS
017632                          64$:
017632  005737  016746          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
017636  001775                          BEQ     1$
017640  005337  016746                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
017644  117766  177102  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
017652  005237  016752                  INC     $TKQOUT         ;;UPDATE THE POINTER
017656  023727  016752  016755          CMP     $TKQOUT,#$TKQEND;;DID IT GO OFF OF THE END?
017664  001003                          BNE     2$              ;;BRANCH IF NO
017666  012737  016754  016752          MOV     #$TKQSRT,$TKQOUT;;RESET THE POINTER
017674  000002                  2$:     RTI                     ;;RETURN
                                                ;;*****************************************************************
                                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                                ;*CALL:
                                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
017676  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
017700  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
017702  012703  020132          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
017706  022703  020144          2$:     CMP     #$TTYIN+10.,R3  ;;BUFFER FULL?
017712  101456                          BLOS    4$              ;;BR IF YES
017714  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
017716  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
017720  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
017724  001022                          BNE     5$              ;;BR IF NO
017726  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
017730  001007                          BNE     6$              ;;BR IF NO
017732  112737  000134  020130          MOVB    #'\,9$          ;;TYPE A BACK SLASH
017740  104401  020130                  TYPE    ,9$
017744  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
017750  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
017752  020327  020132                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
017756  103434                          BLO     4$              ;;BR IF YES
```

```
4314  017760  111337  020130                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
4315  017764  104401  020130                  TYPE    ,9$             ;;GO TYPE
4316  017770  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
4317  017772  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
4318  017774  001406                          BEQ     7$              ;;BR IF NO
4319  017776  112737  000134  020130          MOVB    #'\,9$          ;;TYPE A BACK SLASH
4320  020004  104401  020130                  TYPE    ,9$
4321  020010  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
4322  020012  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
4323  020016  001003                          BNE     8$              ;;BR IF NO
4324  020020  104401  020151                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
4325  020024  000726                          BR      1$              ;;GO START OVER
4326  020026  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
4327  020032  001011                          BNE     3$              ;;BRANCH IF NO
4328  020034  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
4329  020036  104401  001213                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
4330  020042  104401  020132                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
4331  020046  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
4332  020050  104401  001212          4$:     TYPE    ,$QUES          ;;TYPE A '?'
4333  020054  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
4334  020056  111337  020130          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
4335  020062  104401  020130                  TYPE    ,9$
4336  020066  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
4337  020072  001305                          BNE     2$              ;;LOOP IF NOT RETURN
4338  020074  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
4339  020100  104401  001214                  TYPE    ,$LF            ;;TYPE A LINE FEED
4340  020104  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
4341  020106  012603                          MOV     (SP)+,R3        ;;RESTORE R3
4342  020110  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4343  020112  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
4344  020120  012766  020132  000004          MOV     #$TTYIN,4(SP)
4345  020126  000002                          RTI                     ;;RETURN
4346  020130     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
4347  020131     000                          .BYTE   0               ;;TERMINATOR
4348  020132  000012          $TTYIN: .BLKB   10.                     ;;RESERVE 10. BYTES FOR TTY INPUT
4349  020144  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
4350  020151     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
4351  020156  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
4352  020163     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
4353  020170  036440  000040
4354  020174  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
4355  020202  020075     000
4356          020206                  .EVEN
4357
4358                                  .SBTTL  INTEGER DIVIDE ROUTINE
4359
4360                                  ;;*****************************************************************
4361                                  ;*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
4362                                  ;*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
4363                                  ;*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
4364                                  ;*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
4365                                  ;*SAVE SIGN AS THE DIVIDEND.
4366                                  ;*CALL:
4367                                  ;*      MOV     LOW DIVIDEND,-(SP) ;;THE HIGH DIVIDEND MUST BE < 1/2
4368                                  ;*      MOV     HIGH DIVIDEND,-(SP);;    AS LARGE AS THE DIVISOR
4369                                  ;*      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

020206                          $DIV:
020206  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
020210  042716  000017                  BIC     #17,(SP)        ;;STRIP AWAY CONDITION CODES
020214  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020216  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020220  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020222  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020224  005046                          CLR     -(SP)           ;;SAVE A PLACE FOR SIGNS
020226  012746  000021                  MOV     #17.,-(SP)      ;;SETUP THE ITERATION COUNTER
020232  016601  000024                  MOV     24(SP),R1       ;;PICKUP THE DIVIDEND
020236  016600  000022                  MOV     22(SP),R0
020242  100005                          BPL     1$              ;;CHECK THE SIGN
020244  105366  000003                  DECB    3(SP)           ;;KEEP TRACK OF THE SIGN
020250  005400                          NEG     R0              ;;AND NEGATE THE ORIGINAL
020252  005401                          NEG     R1              ;;NUMBER
020254  005600                          SBC     R0
020256  016602  000020          1$:     MOV     20(SP),R2       ;;PICKUP THE DIVISOR
020262  002407                          BLT     2$              ;;CHECK THE SIGN
020264  003011                          BGT     3$              ;;DIVISOR OF 0 IS A NO-NO
020266  052766  000003  000014          BIS     #3,14(SP)       ;;SET "V" & "C"
020274  012700  177777                  MOV     #-1,R0          ;;SET REMAINDER TO ALL ONES
020300  000424                          BR      7$              ;;EXIT
020302  005266  000002          2$:     INC     2(SP)           ;;KEEP TRACK OF DIVISORS SIGN
020306  000401                          BR      4$
020310  005402                  3$:     NEG     R2              ;;NEGATE THE ORIGINAL NUMBER
020312  000241                  4$:     CLC                     ;;CLEAR "C"
020314  000405                          BR      6$              ;;START FORMING QUOTIENT
020316  006100                  5$:     ROL     R0              ;;POSITION MSB'S
020320  010003                          MOV     R0,R3           ;;COPY
020322  060203                          ADD     R2,R3           ;;COMPARE DIVIDEND & DIVISOR
020324  103001                          BCC     6$              ;;BR IF DIVIDEND > DIVISOR
020326  010300                          MOV     R3,R0           ;;REMAINDER AFTER THIS LOOP
020330  006101                  6$:     ROL     R1              ;;QUOTIENT BIT ENTERS HERE
020332  005316                          DEC     (SP)            ;;DONE?
020334  001370                          BNE     5$              ;;BR IF NO
020336  005701                          TST     R1              ;;OVERFLOW?
020340  100005                          BPL     8$              ;;BR IF NO
020342  052766  000002  000014          BIS     #2,14(SP)       ;;SET "V" IN RETURN STATUS WORD
020350  005000                          CLR     R0              ;;SET REMAINDER TO ALL ZEROS
020352  010001                  7$:     MOV     R0,R1           ;;COPY REMAINDER INTO QUOTIENT
020354  005726                  8$:     TST     (SP)+           ;;CLEAR COUNTER FROM STACK
020356  005716                          TST     (SP)            ;;REMAINDER SIGN CORRECTION NEEDED?
020360  002004                          BGE     9$              ;;BR IF NO
```

```
020362  005400                  NEG     R0              ;;NEGATE REMAINDER
020364  105066  000001          CLRB    1(SP)           ;;CLEAR SIGN
020370  005316                  DEC     (SP)            ;;BUT DON'T FORGET QUOTIENT
020372  005726          9$:     TST     (SP)+           ;;QUOTIENT SIGN CORRECTION NEEDED?
020374  001401                  BEQ     10$             ;;BR IF NO
020376  005401                  NEG     R1              ;;NEGATE QUOTIENT
020400  010166  000020  10$:    MOV     R1,20(SP)       ;;RETURN QUOTIENT AND
020404  010066  000016          MOV     R0,16(SP)       ;;REMAINDER TO USER
020410  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
020412  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
020414  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
020416  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
020420  012666  000002          MOV     (SP)+,2(SP)     ;;SETUP TO RETURN CONDITION CODES
020424  000002                  RTI                     ;;RETURN

                        .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                        ;;*****************************************************************
                        ;*SAVE R0-R5
                        ;*CALL:
                        ;*      SAVREG
                        ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                        ;*
                        ;*TOP---(+16)
                        ;* +2---(+18)
                        ;* +4---R5
                        ;* +6---R4
                        ;* +8---R3
                        ;*+10---R2
                        ;*+12---R1
                        ;*+14---R0

020426          $SAVREG:
020426  010046          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020430  010146          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020432  010246          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020434  010346          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020436  010446          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
020440  010546          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020442  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
020446  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
020452  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
020456  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
020462  000002          RTI

                        ;*RESTORE R0-R5
                        ;*CALL:
                        ;*      RESREG

020464          $RESREG:
020464  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
020470  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
020474  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
020500  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
020504  012605          MOV     (SP)+,R5        ;;POP STACK INTO R5
020506  012604          MOV     (SP)+,R4        ;;POP STACK INTO R4
020510  012603          MOV     (SP)+,R3        ;;POP STACK INTO R3
```

```
020512  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
020514  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
020516  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
020520  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

020522                  $RAND:
020522  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020524  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020526  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020530  013700  020622          MOV     $LONUM,R0       ;;SET R0 WITH LOW
020534  013701  020620          MOV     $HINUM,R1       ;;SET R1 WITH HIGH
020540  012702  177771          MOV     #-7,R2          ;;SET SHIFT COUNT
020544  006300                  1$:     ASL     R0      ;;SHIFT R0 LEFT AND
020546  006101                  ROL     R1              ;;ROTATE CARRY INTO R1 AND
020550  005202                  INC     R2              ;;CHECK FOR DONE
020552  001374                  BNE     1$              ;;CONTINUE SHIFT LOOP
020554  063700  020622          ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
020560  005501                  ADC     R1              ;;PROPOGATE CARRY
020562  063701  020620          ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
020566  062700  001057          ADD     #1057,R0        ;;ADD LOW CONSTANT
020572  005501                  ADC     R1              ;;PROPOGATE CARRY
020574  062701  047401          ADD     #47401,R1       ;;ADD HIGH CONSTANT
020600  010037  020622          MOV     R0,$LONUM       ;;SAVE R0
020604  010137  020620          MOV     R1,$HINUM       ;;SAVE R1
020610  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
020612  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
020614  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
020616  000207                  RTS     PC              ;;RETURN
020620  176543          $HINUM: .WORD   176543
020622  123456          $LONUM: .WORD   123456
```

```
                        .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                        ;;*****************************************************************
                        ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                        ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                        ;*POSITIVE.
                        ;*CALL
                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                        ;*      JSR     PC,@#$D82D
                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                        ;*                              ;;IS ON THE STACK

020624  104412          $D82D:  SAVREG                  ;;SAVE REGISTERS
```

```
 4538  020626  016602  000002                 MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
 4539  020632  012700  021004                 MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
 4540  020636  010066  000002                 MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
 4541  020642  012201                         MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
 4542  020644  012202                         MOV     (R2)+,R2
 4543  020646  012737  000012  020722         MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
 4544  020654  012704  020734                 MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
 4545  020660  012705  020736                 MOV     #$TNPWR+2,R5
 4546  020664  005003                 1$:     CLR     R3              ;;CLEAR PARTIAL
 4547  020666  161401                 2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
 4548  020670  005602                         SBC     R2
 4549  020672  161502                         SUB     (R5),R2
 4550  020674  002403                         BLT     3$              ;;BR IF TEN POWER TO LARGE
 4551  020676  005203                         INC     R3              ;;ADD 1 TO PARTIAL
 4552  020700  000772                         BR      2$              ;;LOOP
 4553  020702  062401                 3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
 4554  020704  005502                         ADC     R2
 4555  020706  062402                         ADD     (R4)+,R2
 4556  020710  022525                         CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
 4557  020712  052703  000060                 BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
 4558  020716  110320                         MOVB    R3,(R0)+        ;;SAVE IT
 4559  020720  005327                         DEC     (PC)+           ;;DONE?
 4560  020722  000000                 4$:     .WORD   0
 4561  020724  001357                         BNE     1$              ;;BR IF NO
 4562  020726  105020                         CLRB    (R0)+           ;;TERMINATOR
 4563  020730  104413                         RESREG                  ;;RESTORE REGISTERS
 4564  020732  000207                         RTS     PC              ;;RETURN
 4565  020734  145000                 $TNPWR: 145000                  ;;1.0E09
 4566  020736  035632                         35632
 4567  020740  160400                         160400                  ;;1.0E08
 4568  020742  002765                         2765
 4569  020744  113200                         113200                  ;;1.0E07
 4570  020746  000230                         230
 4571  020750  041100                         041100                  ;;1.0E06
 4572  020752  000017                         17
 4573  020754  103240                         103240                  ;;1.0E05
 4574  020756  000001                         1
 4575  020760  023420                         23420                   ;;1.0E04
 4576  020762  000000                         0
 4577  020764  001750                         1750                    ;;1.0E03
 4578  020766  000000                         0
 4579  020770  000144                         144                     ;;1.0E02
 4580  020772  000000                         0
 4581  020774  000012                         12                      ;;1.0E01
 4582  020776  000000                         0
 4583  021000  000001                         1                       ;;1.0E00
 4584  021002  000000                         0
 4585  021004  000014         $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
 4586
 4587                          .SBTTL  TRAP DECODER
 4588
 4589                          ;;**************************************************************
 4590                          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 4591                          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 4592                          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 4593                          ;*GO TO THAT ROUTINE.
```

```
                                ;
021020  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
021022  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
021026  005740                          TST     -(R0)           ;;BACKUP BY 2
021030  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
021032  006300                          ASL     R0              ;;POSITION FOR INDEXING
021034  016000  021054                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
021040  000200                          RTS     R0              ;;GO TO ROUTINE
                                ;
                                ;
                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
                                ;
021042  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
021044  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
021052  000002                          RTI                     ;;RESTORE THE PSW
                                ;
                                .SBTTL  TRAP TABLE
                                ;
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;
                                ;       ROUTINE
                                ;       -------
021054  021042                  $TRPAD: .WORD   $TRAP2
021056  016054                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
021060  016320                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
021062  016274                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
021064  016334                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
021066  016522                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
                                ;
021070  017334                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
                                ;
021072  017244                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
021074  017606                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
021076  017676                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
021100  020426                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
021102  020464                          $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
                                ;
                                ;;*****************************************************************
                                ;
                                .SBTTL  RP11 HANDLER
                                ;
                                ;;*****************************************************************
                                ;
                                ;STORAGE FOR RPDS & RPCS ON AN ILLEGAL INTERRUPT ERROR
                                        ;RPERRS   = RPDS
                                        ;RPERRS+2 = RPCS
                                ;
021104  000000                  RPERRS: .WORD   0
021106  000000                          .WORD   0
                                ;
                                ;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=1 WORD)
                                        ;THIS IS A ONE WORD QUEUE WHICH WILL CONTAIN A
                                        ;BIT FOR EACH DRIVE WHICH IS ACTIVE WITH A COMMAND.
                                ;
021110  000000                  DRVACT: .WORD   0
```

```
                        ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
                                ;DRVSTA=0 DRIVE NONEXISTENT OR OFFLINE
                                ;DRVSTA>0 DRIVE IS ONLINE
                                ;DRVSTA<0 DRIVE IS UNSAFE OR AN RP03

021112  000            DRVSTA: .BYTE   0               ;DRIVE 0
021113  000                    .BYTE   0               ;DRIVE 1
021114  000                    .BYTE   0               ;DRIVE 2
021115  000                    .BYTE   0               ;DRIVE 3
021116  000                    .BYTE   0               ;DRIVE 4
021117  000                    .BYTE   0               ;DRIVE 5
021120  000                    .BYTE   0               ;DRIVE 6
021121  000                    .BYTE   0               ;DRIVE 7

                        ;DRIVE TYPE INDICATOR (DRVTYP = 1 WORD)
                                ;THIS IS A ONE WORD INDICATOR.  EACH DRIVE IS ASSIGNED ONE BIT
                                ;STARTING AT BIT00 FOR DRIVE 0.  THE DRIVE'S BIT WILL BE SET IF IT
                                ;IS AN RP03 AND RESET IF IT IS AN RP02.

021122  000000          DRVTYP: .WORD   0

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

021124  000000          TRNSWT: .WORD   0

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

021126  000000          SEEKWT: .WORD   0

                        ;RP11 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
                                ;ACTDRV=0 DRIVER IS INACTIVE
                                ;ACTDRV>0 DRIVER IS ACTIVE

021130  000             ACTDRV: .BYTE   0

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

021131  000             ACTSTR: .BYTE   0

                        ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
                                ;SAVEFG<0 SAVE THE RP11E REGISTERS WHEN THE
                                ;OPERATION IS COMPLETED AS PER (DPB+14).
                                ;SAVEFG=0 SAVE THE RP11E REGISTERS, AS PER
                                ;(DPB+14), AFTER AN ERROR.

021132  000000          SAVEFG: .WORD   0
```

```
                                        ;ORDER REQUEST QUEUES/LISTS
                                        ;
021134  000000  000000  000000  QREGS:  .WORD   0,0,0,0,0,0,0,0,0  ;STORE REGISTERS ORDER QUEUE
021142  000000  000000  000000
021150  000000  000000  000000
021156  000000  000000  000000  QSEEK:  .WORD   0,0,0,0,0,0,0,0,0  ;SEEK/HOME SEEK ORDER QUEUE
021164  000000  000000  000000
021172  000000  000000  000000
021200  000000  000000  000000  QTRANS: .WORD   0,0,0,0,0,0,0,0,0  ;TRANSFER ORDER QUEUE
021206  000000  000000  000000
021214  000000  000000  000000
021222  000000  000000  000000  QDONE:  .WORD   0,0,0,0,0,0,0,0,0  ;ON CYLINDER TRANSFER QUEUE
021230  000000  000000  000000
021236  000000  000000  000000
021244  000000  000000  000000  QWAIT:  .WORD   0,0,0,0,0,0,0,0    ;TRANSFER SEEK DONE LIST
021252  000000  000000  000000
021260  000000  000000
                                        ;
                                        ;
                                        ;TIMEOUT TABLE (TIMER=8 WORDS)
                                        ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
                                        ;
021264  177777          TIMER:  .WORD   -1      ;DRIVE 0
021266  177777                  .WORD   -1      ;DRIVE 1
021270  177777                  .WORD   -1      ;DRIVE 2
021272  177777                  .WORD   -1      ;DRIVE 3
021274  177777                  .WORD   -1      ;DRIVE 4
021276  177777                  .WORD   -1      ;DRIVE 5
021300  177777                  .WORD   -1      ;DRIVE 6
021302  177777                  .WORD   -1      ;DRIVE 7
                                        ;
                                        ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
                                        ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
                                        ;ATTENTION BIT
                                        ;
021304  001         ATABIT: .BYTE   1       ;DRIVE 0
021305  002                 .BYTE   2       ;DRIVE 1
021306  004                 .BYTE   4       ;DRIVE 2
021307  010                 .BYTE   10      ;DRIVE 3
021310  020                 .BYTE   20      ;DRIVE 4
021311  040                 .BYTE   40      ;DRIVE 5
021312  100                 .BYTE   100     ;DRIVE 6
021313  200                 .BYTE   200     ;DRIVE 7
                                        ;
                                        ;RP11E DRIVER INITIALIZATION CODE
                                        ;THIS ROUTINE WILL DETERMINE WHICH RP02 OR RP03 DRIVES ARE
                                        ;AVAILABLE FOR TESTING AND SET THE 'DRVSTA' INDICATOR
                                        ;TO THE PROPER STATE FOR EACH DRIVE.
                                        ;NOTE: THIS ROUTINE CALLS DRVINT
                                        ;
                                        ;CALL
                                        ;
                                        ;       JSR     PC,RPINIT
                                        ;       RETURN
                                        ;
021314  104412          RPINIT: SAVREG                  ;SAVE R0-R5
```

```
 4762  021316  013746  177776                 MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
 4763  021322  013737  001222  177776         MOV     RPPRIO,@#PS     ;CHANGE THE PRIORITY
 4764  021330  012701  021104                 MOV     #RPERRS,R1      ;FIRST ADDRESS TO BE CLEARED
 4765  021334  012702  021262                 MOV     #TIMER-2,R2     ;LAST ADDRESS TO BE CLEARED
 4766  021340  005021                 1$:     CLR     (R1)+           ;CLEAR
 4767  021342  020102                         CMP     R1,R2           ;ARE WE DONE?
 4768  021344  101775                         BLOS    1$              ;BRANCH IF NO
 4769  021346  012702  021302                 MOV     #ATABIT-2,R2    ;LAST ADDRESS
 4770  021352  012721  177777         2$:     MOV     #-1,(R1)+       ;INITIALIZE
 4771  021356  020102                         CMP     R1,R2           ;DONE?
 4772  021360  101774                         BLOS    2$              ;LOOP IF NO
 4773  021362  013703  001220                 MOV     RPVEC,R3        ;SETUP THE RP11 VECTOR
 4774  021366  012723  022502                 MOV     #ISR,(R3)+
 4775  021372  013713  001222                 MOV     RPPRIO,(R3)
 4776  021376  013704  001216                 MOV     RPADR,R4        ;RP11 ADDRESS
 4777  021402  005001                         CLR     R1              ;START WITH DRIVE 0
 4778  021404  004737  021444         3$:     JSR     PC,DRVINT       ;INIT THE DRIVE
 4779  021410  005201         4$:     INC     R1                      ;GO TO NEXT DRIVE
 4780  021412  042701  177770                 BIC     #^C7,R1         ;MASK OUT UNUSED BITS
 4781  021416  001372                         BNE     3$              ;BR IF MORE DRIVES TO GO
 4782  021420  012701  000007                 MOV     #7,R1           ;START WITH DRIVE 7
 4783  021424  005037  177776                 CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
 4784  021430  005301         5$:     DEC     R1                      ;GO TO THE NEXT DRIVE
 4785  021432  100376                         BPL     5$              ;CHECK NEXT DRIVE
 4786  021434  012637  177776                 MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
 4787  021440  104413                         RESREG                  ;RESTORE R0-R5
 4788  021442  000207                         RTS     PC              ;BYE-BYE
 4789
 4790                                 ;DRIVE INITIALIZATION ROUTINE
 4791                                         ;THIS ROUTINE DETERMINES OF A DRIVE IS PRESENT AND IF IT IS
 4792                                         ;AN RP02 OR RP03.  'DRVSTA' IS SET TO REFLECT THE DRIVE'S
 4793                                         ;STATUS AND THE DRIVE TYPE INDICATOR BIT IS SET IN 'DRVTYP'.
 4794                                 ;CALL
 4795                                 ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 4796                                 ;       MOV     RPADR,R4        ;UNIBUS ADDRESS OF RP11 (RPDS)
 4797                                 ;       JSR     PC,DRVINT       ;CALLED BY A JSR
 4798                                 ;       RETURN                  ;NORMAL RETURN
 4799                                 ;
 4800  021444  104412                 DRVINT: SAVREG                  ;SAVE R0-R5
 4801  021446  112761  177777  021112         MOVB    #-1,DRVSTA(R1)  ;START DRIVE STATUS AS UNSAFE
 4802  021454  110164  000005                 MOVB    R1,RPCS+1(R4)   ;SELECT A DRIVE
 4803  021460  032764  001000  000000         BIT     #SUFU,RPDS(R4)  ;DRIVE UNSAFE ?
 4804  021466  001022                         BNE     3$              ;BR IF UNSAFE
 4805  021470  032764  040000  000000         BIT     #SUOL,RPDS(R4)  ;DRIVE ONLINE ?
 4806  021476  001410                         BEQ     1$              ;BR IF NOT
 4807  021500  032764  004000  000000         BIT     #SUSI,RPDS(R4)  ;SEEK INCOMPLETE SET ?
 4808  021506  001007                         BNE     2$              ;BR IF SET
 4809  021510  032764  100000  000000         BIT     #SURDY,RPDS(R4) ;DRIVE READY ?
 4810  021516  001003                         BNE     2$              ;BR IF READY
 4811  021520  105061  021112         1$:     CLRB    DRVSTA(R1)      ;SET DRIVE OFFLINE OR NONEXISTENT
 4812  021524  000416                         BR      5$              ;EXIT
 4813  021526  112761  000001  021112 2$:     MOVB    #1,DRVSTA(R1)   ;SET DRIVE AVAILABLE
 4814  021534  032764  020000  000000 3$:     BIT     #SURP03,RPDS(R4)  ;DRIVE AN RP03 ?
 4815  021542  001004                         BNE     4$              ;BR IF IT IS
 4816  021544  146137  021304  021122         BICB    ATABIT(R1),DRVTYP  ;SET RP02 INDICATOR
 4817  021552  000403                         BR      5$              ;EXIT
```

```
                                        4$:     BISB    ATABIT(R1),DRVTYP  ;SET RP03 INDICATOR
                                        5$:     RESREG                     ;RESTORE R0-R5
                                                RTS     PC                 ;RETURN

                                ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
                                ;
                                ;CALL
                                ;
                                ;       JSR     R0,@#RP11          ;CALL THE RP11E DRIVER
                                ;       PNTADR                     ;ADDRESS OF POINTER OF DRIVE'S PARAMETER BLOCK
                                ;       RETURN1                    ;RETURN HERE IF QUEUE IS FULL
                                ;       RETURN2                    ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
                                                                   ;IS AN ERROR CONDITION

                                RP11:   MOV     PSW,-(SP)          ;SAVE THE CALLING STATUS
                                        MOV     RPPRIO,PSW         ;DON'T ALLOW ANY RP11E INTERRUPTS
                                        MOVB    #1,ACTDRV          ;SET "ACTIVE DRIVER" FLAG
                                        MOV     R2,-(SP)           ;SAVE R2
                                        MOV     (R0)+,R2           ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
                                        CLR     16(R2)             ;CLEAR THE STATUS/ERROR INDICATOR
                                        JSR     R0,LODQUE          ;PUT THIS REQUEST IN QUEUE
                                        BR      1$                 ;QUEUE IS FULL
                                        JSR     PC,CI              ;INITIATE A COMMAND
                                        TST     (R0)+
                                1$:     MOV     (SP)+,R2           ;RESTORE R2
                                        CLRB    ACTDRV             ;CLEAR "ACTIVE DRIVER" FLAG
                                        MOV     (SP)+,@#PS         ;RETURN "PS" TO USER LEVEL
                                        RTS     R0                 ;RETURN TO CALLER

                                ;COMMAND INITIATION ROUTINE

                                CI:     SAVREG                     ;SAVE R0-R5
                                        MOV     RPADR,R4           ;RP11 ADDRESS
                                        TST     TRNSWT             ;TRANSFER IN PROGRESS ?
                                        BNE     5$                 ;BR IF ONE IS
                                        TST     QREGS              ;PENDING STORE REGISTERS OP ?
                                        BEQ     1$                 ;BR IF NOT
                                        JSR     PC,CI1             ;STORE THE REGISTERS
                                1$:     TST     QSEEK              ;PENDING SEEKS OR HOME SEEKS ?
                                        BEQ     2$                 ;BR IF NONE
                                        JSR     PC,CI2             ;START PENDING SEEKS
                                2$:     TST     QTRANS             ;PENDING TRANSFER ORDERS ?
                                        BEQ     3$                 ;BR IF NONE
                                        JSR     PC,CI3             ;START THE TRANSFER ORDERS
                                3$:     TST     QDONE              ;ON CYLINDER TRANSFER REQUESTS ?
                                        BEQ     4$                 ;BR IF DONE
                                        JSR     PC,CI4             ;START ORDER AT HEAD OF QUEUE
                                        TST     TRNSWT             ;TRANSFER ORDER STARTED ?
                                        BNE     5$                 ;BR IF ONE WAS
                                4$:     TSTB    DRVACT             ;ANY ACTIVE DRIVES ?
                                        BEQ     5$                 ;BR IF NONE
                                        MOVB    #40,RPCS+1(R4)     ;SET ATTENTION INTERRUPT ENABLE
                                5$:     RESREG                     ;RESTORE R0-R5
                                        RTS     PC                 ;RETURN

                                ;STORE THE REGISTERS COMMAND
```

```
4874
4875  021760  117764  177150  000005  CI1:    MOVB    @QREGS,RPCS+1(R4)  ;SELECT THE DRIVE
4876  021766  013702  021134          MOV     QREGS,R2           ;DPB ADDRESS
4877  021772  004737  024000          JSR     PC,SVRP11          ;STORE THE REGISTERS
4878  021776  012762  000200  000016  MOV     #BIT07,16(R2)      ;SET 'DONE'
4879  022004  012702  021134          MOV     #QREGS,R2          ;QUEUE ADDRESS
4880  022010  004737  024256          JSR     PC,POPQUE          ;'POP' THE QUEUE
4881  022014  005737  021134          TST     QREGS              ;ANY MORE REQUESTS IN QUEUE ?
4882  022020  001357                  BNE     CI1                ;BR IF ANY ARE
4883  022022  000207                  RTS     PC                 ;RETURN
4884
4885                          ;START THE SEEK OR HOME SEEK COMMAND
4886
4887  022024  117764  177126  000005  CI2:    MOVB    @QSEEK,RPCS+1(R4)  ;SELECT THE DRIVE
4888  022032  013702  021156          MOV     QSEEK,R2           ;DPB ADDRESS
4889  022036  005001                  CLR     R1                 ;CLEAR R1
4890  022040  117701  177112          MOVB    @QSEEK,R1          ;DRIVE ADDRESS
4891  022044  004037  022436          JSR     R0,CIR             ;SEE IF DRIVE IS READY
4892  022050  000424                  BR      1$                 ;DRIVE NOT READY OR IS UNSAFE
4893  022052  016264  000010  000012  MOV     10(R2),RPCA(R4)    ;DESIRED CYLINDER ADDRESS
4894  022060  016264  000012  000014  MOV     12(R2),RPDA(R4)    ;SECTOR/TRACK ADDRESS
4895  022066  116264  000002  000004  MOVB    2(R2),RPCS(R4)     ;START THE COMMAND
4896  022074  156137  021304  021110  BISB    ATABIT(R1),DRVACT  ;SET THE DRIVE'S ACTIVE BIT
4897  022102  006301                  ASL     R1                 ;ADDRESS WORDS
4898  022104  013761  001260  021264  MOV     TIMOUT,TIMER(R1)   ;START TIMER FOR DRIVE
4899  022112  013761  021156  021244  MOV     QSEEK,QWAIT(R1)    ;PUT DPB ADDRESS IN 'WAIT' QUEUE
4900  022120  006201                  ASR     R1                 ;RESTORE DRIVE ADDRESS
4901  022122  012701  021156          1$:     MOV     #QSEEK,R1  ;QUEUE ADDRESS
4902  022126  004737  024256          JSR     PC,POPQUE          ;'POP' THE SEEK QUEUE
4903  022132  005737  021156          TST     QSEEK              ;MORE REQUESTS ?
4904  022136  001332                  BNE     CI2                ;BR IF MORE
4905  022140  000207                  RTS     PC                 ;RETURN
4906
4907                          ;DO AN EXPLICIT SEEK IF DRIVE IS NOT ON CYLINDER
4908
4909  022142  117764  177032  000005  CI3:    MOVB    @QTRANS,RPCS+1(R4)  ;SELECT THE DRIVE
4910  022150  005001                  CLR     R1                 ;CLEAR R1
4911  022152  117701  177022          MOVB    @QTRANS,R1         ;DRIVE ADDRESS
4912  022156  013702  021200          MOV     QTRANS,R2          ;DPB ADDRESS
4913  022162  004037  022436          JSR     R0,CIR             ;SEE IF DRIVE READY
4914  022166  000426                  BR      1$                 ;DRIVE NOT READY
4915  022170  016264  000010  000012  MOV     10(R2),RPCA(R4)    ;DESIRED CYLINDER
4916  022176  016264  000012  000014  MOV     12(R2),RPDA(R4)    ;SECTOR/TRACK ADDRESS
4917  022204  112764  000011  000004  MOVB    #SEEK,RPCS(R4)     ;SEEK COMMAND
4918  022212  156137  021304  021110  BISB    ATABIT(R1),DRVACT  ;SET DRIVE ACTIVE
4919  022220  156137  021304  021126  BISB    ATABIT(R1),SEEKWT  ;SET SEEK WAIT BIT
4920  022226  006301                  ASL     R1                 ;SETUP TO ADDRESS WORDS
4921  022230  013761  001260  021264  MOV     TIMOUT,TIMER(R1)   ;START TIMER FOR DRIVE
4922  022236  013761  021200  021244  MOV     QTRANS,QWAIT(R1)   ;'WAIT' TABLE ENTRY
4923  022244  012701  021200          1$:     MOV     #QTRANS,R1 ;TRANSFER QUEUE ADDRESS
4924  022250  004737  024256          JSR     PC,POPQUE          ;'POP' THE TRANSFER QUEUE
4925  022254  005737  021200          TST     QTRANS             ;MORE DRIVES IN QUEUE ?
4926  022260  001330                  BNE     CI3                ;BR IF MORE DRIVES
4927  022262  000207                  RTS     PC                 ;RETURN
4928
4929                          ;START THE PENDING ON CYLINDER TRANSFER REQUEST
```

```
4930
4931  022264  117764  176732  000005  CI4:    MOVB    @QDONE,RPCS+1(R4)  ;SELECT THE DRIVE
4932  022272  117701  176724                  MOVB    @QDONE,R1          ;DRIVE ADDRESS
4933  022276  013702  021222                  MOV     QDONE,R2           ;DPB ADDRESS
4934  022302  004037  022436                  JSR     R0,CIR             ;SEE IF DRIVE IS READY
4935  022306  000440                          BR      1$                 ;DRIVE NOT READY
4936  022310  016264  000004  000006          MOV     4(R2),RPWC(R4)     ;WORD COUNT
4937  022316  016264  000006  000010          MOV     6(R2),RPBA(R4)     ;BUFFER ADDRESS
4938  022324  016264  000010  000012          MOV     10(R2),RPCA(R4)    ;CYLINDER ADDRESS
4939  022332  016264  000012  000014          MOV     12(R2),RPDA(R4)    ;SECTOR/TRACK ADDRESS
4940  022340  016246  000002                  MOV     2(R2),-(SP)        ;COMMAND CODE, MODE & HDR BITS
4941  022344  151266  000001                  BISB    (R2),1(SP)         ;DRIVE ADDRESS
4942  022350  156216  000001                  BISB    1(R2),(SP)         ;'MEX' BITS
4943  022354  052716  000100                  BIS     #IDE,(SP)          ;INTERRUPT ON DONE BIT
4944  022360  012664  000004                  MOV     (SP)+,RPCS(R4)     ;START THE COMMAND
4945  022364  156137  021304  021110          BISB    ATABIT(R1),DRVACT  ;DRIVE ACTIVE BIT
4946  022372  006301                          ASL     R1                 ;ADDRESS WORDS
4947  022374  013761  001260  021264          MOV     TIMOUT,TIMER(R1)   ;START TIMER FOR DRIVE
4948  022402  006201                          ASR     R1                 ;RESTORE R1
4949  022404  010237  021124                  MOV     R2,TRNSWT          ;DPB ADDRESS
4950  022410  012701  021222          1$:     MOV     #QDONE,R1          ;QUEUE ADDRESS
4951  022414  004737  024256                  JSR     PC,POPQUE          ;'POP' THE QUEUE
4952  022420  005737  021124                  TST     TRNSWT             ;TRANSFER ORDER RUNNING ?
4953  022424  001003                          BNE     2$                 ;BR IF ONE IS
4954  022426  005737  021222                  TST     QDONE              ;MORE ORDERS WAITING ?
4955  022432  001314                          BNE     CI4                ;BR IF MORE
4956  022434  000207          2$:     RTS     PC                         ;RETURN
4957
4958                          ;SEE IF DRIVE IS ONLINE
4959
4960  022436  004737  021444          CIR:    JSR     PC,DRVINT          ;CHECK DRIVE'S STATUS
4961  022442  105761  021112                  TSTB    DRVSTA(R1)         ;SEE WHAT DRIVE'S STATUS IS
4962  022446  003013                          BGT     3$                 ;BR IF DRIVE ONLINE
4963  022450  002404                          BLT     1$                 ;BR IF DRIVE IS UNSAFE
4964  022452  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE INDICATOR
4965  022460  000403                          BR      2$                 ;GO STORE REGISTERS
4966  022462  012762  110000  000016  1$:     MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE INDICATOR
4967  022470  004737  024000          2$:     JSR     PC,SVRP11          ;STORE THE REGISTERS
4968  022474  000401                          BR      4$                 ;EXIT
4969  022476  005720          3$:     TST     (R0)+              ;STEP AROUND ERROR RETURN
4970  022500  000200          4$:     RTS     R0                 ;RETURN
4971
4972                          ;INTERRUPT SERVICE ROUTINE
4973
4974  022502  112737  000001  021130  ISR:    MOVB    #1,ACTDRV          ;SET "ACTIVE DRIVER" FLAG
4975  022510  104412                          SAVREG                     ;SAVE R0-R5
4976  022512  013704  001216                  MOV     RPADR,R4           ;ADDRESS OF RP11
4977  022516  005737  021124                  TST     TRNSWT             ;TRANSFER OPERATION IN PROGRESS ?
4978  022522  001403                          BEQ     1$                 ;BR IF NOT
4979  022524  004737  022552                  JSR     PC,TD              ;CALL TRANSFER DONE
4980  022530  000402                          BR      2$                 ;EXIT
4981  022532  004737  023034          1$:     JSR     PC,SC              ;CALL SPECIAL CONDITIONS
4982  022536  004737  021646          2$:     JSR     PC,CI              ;GO TO THE COMMAND INITIATION ROUTINE
4983  022542  104413                          RESREG                     ;RESTORE R0-R5
4984  022544  105037  021130                  CLRB    ACTDRV             ;CLEAR "ACTIVE DRIVER" FLAG
4985  022550  000002                          RTI                        ;RETURN
```

```
                                        ;TRANSFER DONE ROUTINE
                                        
022552  005001                  TD:     CLR     R1                      ;CLEAR DRIVE NUMBER STORAGE
022554  117701  176344                  MOVB    @TRNSWT,R1              ;GET DRIVE NUMBER
022560  146137  021304  021110          BICB    ATABIT(R1),DRVACT       ;CLEAR DRIVE ACTIVE BIT
022566  006301                          ASL     R1
022570  012761  177777  021264          MOV     #-1,TIMER(R1)           ;STOP TIMER
022576  006201                          ASR     R1
022600  013702  021124                  MOV     TRNSWT,R2               ;GET "DPB" ADDRESS FROM THE
022604  005037  021124                  CLR     TRNSWT                  ;TRANSFER WAIT QUEUE--CLEAR QUEUE
022610  012762  000200  000016          MOV     #BIT07,16(R2)           ;SET DONE
022616  110164  000005                  MOVB    R1,RPCS+1(R4)           ;SELECT THE DRIVE
022622  116164  021304  000000          MOVB    ATABIT(R1),RPDS(R4)     ;CLEAR DRIVE'S ATTENTION BIT
022630  005764  000004                  TST     RPCS(R4)                ;ERROR SET ?
022634  100413                          BMI     2$                      ;BR IF YES
022636  005737  021132                  TST     SAVEFG                  ;SAVE THE RP11E REGISTERS?
022642  100002                          BPL     1$                      ;BRANCH IF NO
022644  004737  024000                  JSR     PC,SVRP11               ;YES--SAVE THE REGISTERS
022650  042764  000100  000004  1$:     BIC     #IDE,RPCS(R4)           ;CLEAR 'INTERRUPT ON DONE' BIT
022656  004737  022742                  JSR     PC,WC                   ;SEE IF WRITE CHECK TO BE PUT IN QUEUE
022662  000505                          BR      SCD                     ;SPECIAL CONDITIONS ENTRY
022664  004737  024000          2$:     JSR     PC,SVRP11               ;SAVE THE REGISTERS
022670  012764  000001  000004          MOV     #CLEAR,RPCS(R4)         ;CLEAR THE CONTROLLER
022676  004737  021444                  JSR     PC,DRVINT               ;CHECK THE DRIVE'S STATUS
022702  105761  021112                  TSTB    DRVSTA(R1)              ;DRIVE STILL ONLINE ?
022706  003011                          BGT     4$                      ;BR IF ONLINE
022710  100404                          BMI     3$                      ;BR IF UNSAFE
022712  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)     ;SET DRIVE OFFLINE INDICATOR
022720  000466                          BR      SCD                     ;CHECK FOR OTHER DRIVES
022722  012762  110000  000016  3$:     MOV     #BIT15!BIT12,16(R2)     ;SET DRIVE UNSAFE INDICATOR
022730  000462                          BR      SCD                     ;CHECK FOR OTHER DRIVES
022732  052762  100100  000016  4$:     BIS     #BIT15!BIT06,16(R2)     ;SET DATA ERROR FLAG
022740  000456                          BR      SCD                     ;SPECIAL CONDITIONS ENTRY

                                        ;FORCED WRITE CHECK ROUTINE

022742  005737  001322          WC:     TST     AUTOCK                  ;AUTOMATIC WRITE CHECKS ?
022746  001430                          BEQ     1$                      ;BR IF NOT
022750  132762  000004  000002          BITB    #BIT02,$COMND(R2)       ;WRITE COMMAND ?
022756  001024                          BNE     1$                      ;BR IF NOT
022760  010146                          MOV     R1,-(SP)                ;SAVE R1
022762  012701  021200                  MOV     #QTRANS,R1              ;TRANSFER QUEUE ADDRESS
022766  004037  024226                  JSR     R0,DRVQUE               ;PUT THE OPERATION IN THE QUEUE
022772  000416                          BR      1$                      ;QUEUE IS FULL
022774  005062  000016                  CLR     16(R2)                  ;CLEAR 'DONE' BIT IN DPB
023000  116262  000002  000024          MOVB    $COMND(R2),$PREVO(R2)   ;SAVE WRITE OPERATION CODE
023006  016262  000010  000032          MOV     $CYL(R2),$PREVA+2(R2)   ;SAVE CYLINDER
023014  016262  000012  000030          MOV     $SEC(R2),$PREVA(R2)     ;SAVE SECTOR AND TRACK ADDRESSES
023022  112762  000007  000002          MOVB    #WRTCHK,$COMND(R2)      ;CHANGE DRIVER CODE TO WRITE CHECK
023030  012601                  1$:     MOV     (SP)+,R1                ;RESTORE R1
023032  000207                          RTS     PC                      ;EXIT

                                        ;SPECIAL CONDITION ROUTINE

023034  105764  000000          SC:     TSTB    RPDS(R4)                ;ATTENTION BITS SET ?
```

```
5042  023040  001016                        BNE     SCO                     ;BR IF SOME SET
5043  023042  016437  000000  021104        MOV     RPDS(R4),RPERRS         ;STORE DRIVE STATUS REGITER
5044  023050  016437  000004  021106        MOV     RPCS(R4),RPERRS+2       ;STORE THE CONTROL REGISTER
5045  023056  104003                        ERROR   3                       ;ILLEGAL INTERRUPT
5046  023060  005737  021110                TST     DRVACT                  ;ANY DRIVES ACTIVE ?
5047  023064  001403                        BEQ     1$                      ;BR IF NOT
5048  023066  112764  000040  000005        MOVB    #40,RPCS+1(R4)          ;SET 'AIE'
5049  023074  000207                1$:     RTS     PC                      ;RETURN
```

```
5050  023076  005001                  SC0:    CLR     R1              ;CLEAR DRIVE NUMBER STORAGE
5051  023100  012703  000001                  MOV     #1,R3           ;START WITH DRIVE 0'S BIT
5052  023104  030337  021110          SC1:    BIT     R3,DRVACT       ;IS THIS DRIVE ACTIVE ?
5053  023110  001004                          BNE     SC3             ;BR IF IT IS
5054  023112  005201                  SC2:    INC     R1              ;GO TO NEXT DRIVE
5055  023114  106303                          ASLB    R3              ;SHIFT DRIVE'S BIT
5056  023116  001372                          BNE     SC1             ;BR IF MORE DRIVES TO CHECK
5057  023120  000207                          RTS     PC              ;RETURN
5058  023122  006301                  SC3:    ASL     R1              ;SETUP TO ADDRESS WORDS
5059  023124  016102  021244                  MOV     QWAIT(R1),R2    ;GET DRIVE'S DPB ADDRESS
5060  023130  006201                          ASR     R1              ;RESTORE ADDRESS
5061  023132  110164  000005                  MOVB    R1,RPCS+1(R4)   ;SELECT DRIVE
5062  023136  032764  002000  000000          BIT     #SUSU,RPDS(R4)  ;DRIVE STILL SEEKING ?
5063  023144  001362                          BNE     SC2             ;BR IF IT IS
5064  023146  006301                          ASL     R1              ;ADDRESS WORDS
5065  023150  005061  021244                  CLR     QWAIT(R1)       ;CLEAR THE WAIT QUEUE ENTRY
5066  023154  012761  177777  021264          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
5067  023162  006201                          ASR     R1              ;RESTORE DRIVE ADDRESS
5068  023164  146137  021304  021110          BICB    ATABIT(R1),DRVACT  ;CLEAR THE DRIVE'S ACTIVE BIT
5069  023172  116164  021304  000000          MOVB    ATABIT(R1),RPDS(R4)  ;CLEAR DRIVE'S ATTENTION BIT
5070  023200  032764  001000  000000          BIT     #SUFU,RPDS(R4)  ;DRIVE UNSAFE ?
5071  023206  001027                          BNE     1$              ;BR IF IT IS
5072  023210  032764  040000  000000          BIT     #SUOL,RPDS(R4)  ;IS DRIVE STILL ONLINE ?
5073  023216  001432                          BEQ     2$              ;BR IF NOT
5074  023220  032764  140000  000004          BIT     #ERR!HE,RPCS(R4)  ;ANY ERRORS SET ?
5075  023226  001034                          BNE     3$              ;BR IF SET
5076  023230  136137  021304  021126          BITB    ATABIT(R1),SEEKWT  ;TRANSFER SEEK ?
5077  023236  001434                          BEQ     4$              ;BR IF NOT
5078  023240  010146                          MOV     R1,-(SP)        ;SAVE R1
5079  023242  012701  021222                  MOV     #QDONE,R1       ;DONE QUEUE ADDRESS
5080  023246  004037  024226                  JSR     R0,DRVQUE       ;PUT DRIVE'S DPB ADDRESS IN DONE QUEUE
5081  023252  000240                          NOP                     ;RETURN IF QUEUE FULL (SHOULDN'T HAPPEN)
5082  023254  012601                          MOV     (SP)+,R1        ;RESTORE R1
5083  023256  146137  021304  021126          BICB    ATABIT(R1),SEEKWT  ;CLEAR WAIT BIT
5084  023264  000712                          BR      SC2             ;GET ANOTHER DRIVE
5085  023266  052762  110000  000016  1$:     BIS     #BIT15!BIT12,16(R2)  ;SET DRIVE UNSAFE STATUS
5086  023274  112761  177777  021112          MOVB    #-1,DRVSTA(R1)  ;SET DRIVE UNSAFE
5087  023302  000420                          BR      5$              ;STORE REGISTERS
5088  023304  052762  140000  000016  2$:     BIS     #BIT15!BIT14,16(R2)  ;SET DRIVE OFFLINE STATUS
5089  023312  105061  021112                  CLRB    DRVSTA(R1)      ;SET DRIVE OFFLINE
5090  023316  000412                          BR      5$              ;STORE REGISTERS
5091  023320  052762  100240  000016  3$:     BIS     #BIT15!BIT7!BIT5,16(R2)  ;SET ERROR DURING SEEK STATUS
5092  023326  000406                          BR      5$              ;STORE REGISTERS
5093  023330  052762  000200  000016  4$:     BIS     #BIT07,16(R2)   ;NORMAL END OF ORDER STATUS
5094  023336  005737  021132                  TST     SAVEFG          ;SAVE REGISTERS ?
5095  023342  001663                          BEQ     SC2             ;BR IF NOT
5096  023344  146137  021304  021126  5$:     BICB    ATABIT(R1),SEEKWT  ;CLEAR DRIVE'S WAIT BIT
5097  023352  004737  024000                  JSR     PC,SVRP11       ;STORE THE RP11E REGISTERS
5098  023356  000655                          BR      SC2             ;GET ANOTHER DRIVE
5099
5100                                  ;TIMER ROUTINE
5101                                  ;CALL
5102                                  ;       MOV     #TIME,-(SP)     ;ELASPED TIME IN MILLISECONDS ON THE STACK
5103                                  ;       JSR     PC,RPTMR        ;CALL RP11E TIMER ROUTINE
5104
5105  023360  005737  021130          RPTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
```

```
                                                        BNE     5$              ;IF NON-ZERO, EXIT
                                                        MOVB    #1,ACTSTR       ;SET "ACTSTR"
                                                        SAVREG                  ;SAVE R0-R5
                                                        CLR     R1              ;START WITH DRIVE 0
                                                        CLR     R3
                                                1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
                                                        BLT     2$              ;BRANCH IF NO
                                                        SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
                                                        BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
                                                        JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
                                                        BR      3$              ;GO TO THE EXIT
                                                2$:     INC     R1              ;MOVE TO NEXT DRIVE
                                                        TST     (R3)+
                                                        CMP     #8.,R1          ;OUT OF DRIVES?
                                                        BGT     1$              ;BRANCH IF NO
                                                        BR      4$              ;BYPASS 'SC' AND 'CI' CHECKS
                                                3$:     TST     TRNSWT          ;ACTIVE TRANSFER COMMAND ?
                                                        BNE     4$              ;BR IF YES
                                                        JSR     PC,SCO          ;LOOK FOR DRIVES FINISHED SEEKING
                                                        JSR     PC,CI           ;START ANY OUTSTANDING COMMANDS
                                                4$:     RESREG                  ;RESTORE R0-R5
                                                        CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
                                                5$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
                                                        RTS     PC              ;RETURN

                                                ;SOFTWARE TIMEOUT ROUTINE
                                                ;
                                                ;CALL:  STO
                                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                                ;       JSR     PC,STO          ;CALL--DRVACT MUST BE NONZERO
                                                ;       RETURN

                                                STO:    MOV     RPADR,R4        ;GET ADDRESS OF RP11
                                                        MOVB    R1,RPCS+1(R4)   ;SELECT THE DRIVE
                                                        ASL     R1              ;WORD INDEX
                                                        MOV     #-1,TIMER(R1)   ;STOP TIMER
                                                        ASR     R1              ;RESTORE R1
                                                        TST     TRNSWT          ;DATA TRANSFER IN PROGRESS ?
                                                        BEQ     STO1            ;BR IF NOT
                                                        CMPB    R1,@TRNSWT      ;IN PROGRESS FOR THIS DRIVE ?
                                                        BNE     STO5            ;BR IF NOT
                                                        TSTB    RPCS(R4)        ;CONTROLLER READY ?
                                                        BMI     STO5            ;BR IF READY
                                                        BICB    ATABIT(R1),DRVACT  ;CLEAR DRIVE ACTIVE BIT
                                                        MOV     TRNSWT,R2       ;PUT DPB ADDRESS IN R2
                                                        JSR     PC,SVRP11       ;STORE THE REGISTERS
                                                        JSR     PC,DRVINT       ;CHECK ON THE DRIVE'S STATUS
                                                        MOV     #BIT15!BIT08,16(R2)  ;SET TIMEOUT & 'RESET' INDICATOR
                                                        TSTB    DRVSTA(R1)      ;SEE IF DRIVE STILL ONLINE
                                                        BGT     2$              ;BR IF DRIVE STILL ONLINE
                                                        BLT     1$              ;BR IF DRIVE UNSAFE
                                                        BIS     #BIT14,16(R2)   ;SET OFFLINE INDICATOR
                                                        BR      2$              ;GO INIT THE DRIVE
                                                1$:     BIS     #BIT12,16(R2)   ;SET UNSAFE INDICATOR
                                                2$:     MOVB    #CLEAR,RPCS(R4) ;CLEAR THE CONTROLLER
                                                        MOVB    #CLEAR,RPCS(R4) ;CLEAR THE CONTROLLER AGAIN
```

```
5162  023630  005037  021124            CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
5163  023634  000460                    BR      ST05            ;EXIT
5164  023636  006301            ST01:   ASL     R1              ;ADDRESS WORDS
5165  023640  016102  021244            MOV     QWAIT(R1),R2    ;GET DPB ADDRESS
5166  023644  006201                    ASR     R1              ;RESTORE R1
5167  023646  032764  001000  000000    BIT     #SUFU,RPDS(R4)  ;DRIVE UNSAFE ?
5168  023654  001020                    BNE     ST02            ;BR IF UNSAFE
5169  023656  032764  004000  000000    BIT     #SUSI,RPDS(R4)  ;SEEK INCOMPLETE ?
5170  023664  001044                    BNE     ST05            ;BR IF SET
5171  023666  032764  040000  000000    BIT     #SUOL,RPDS(R4)  ;DRIVE ONLINE ?
5172  023674  001417                    BEQ     ST03            ;BR IF NOT
5173  023676  032764  100000  000000    BIT     #SURDY,RPDS(R4) ;DRIVE READY ?
5174  023704  001034                    BNE     ST05            ;BR IF READY
5175  023706  012762  101000  000016    MOV     #BIT15!BIT09,16(R2)  ;SET INDICATOR FOR TIMEOUT DURING SEEK
5176  023714  000414                    BR      ST04            ;CONTINUE WITH PROCESSING
5177  023716  012762  110000  000016    ST02:   MOV     #BIT15!BIT12,16(R2)  ;DRIVE UNSAFE INDICATOR
5178  023724  112761  177777  021112            MOVB    #-1,DRVSTA(R1)  ;SET DRIVE UNSAFE
5179  023732  000405                    BR      ST04            ;CONTINUE
5180  023734  012762  140000  000016    ST03:   MOV     #BIT15!BIT14,16(R2)  ;DRIVE OFFLINE INDICATOR
5181  023742  105061  021112            CLRB    DRVSTA(R1)      ;SET DRIVE OFFLINE
5182  023746  006301            ST04:   ASL     R1              ;ADDRESS WORDS
5183  023750  005061  021244            CLR     QWAIT(R1)       ;CLEAR THE ENTRY
5184  023754  006201                    ASR     R1              ;RESTORE R1
5185  023756  146137  021304  021110    BICB    ATABIT(R1),DRVACT  ;CLEAR DRIVE'S ACTIVE BIT
5186  023764  146137  021304  021126    BICB    ATABIT(R1),SEEKWT  ;CLEAR SEEK WAIT BIT
5187  023772  004737  024000            JSR     PC,SVRP11       ;SAVE THE REGISTERS
5188  023776  000207            ST05:   RTS     PC              ;RETURN
5189
5190
5191                                    ;ROUTINE TO SAVE THE RP11E REGISTERS AS PER DPB+14
5192                                    ;
5193                                    ;CALL
5194                                    ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
5195                                    ;       JSR     PC,SVRP11       ;SAVE THE DRIVES REG'S
5196
5197  024000  104412            SVRP11: SAVREG                  ;SAVE R0-R5
5198  024002  013704  001216            MOV     RPADR,R4        ;RP11 ADDRESS
5199  024006  111264  000005            MOVB    (R2),RPCS+1(R4) ;SELECT DRIVE
5200  024012  016202  000014            MOV     14(R2),R2       ;GET THE ERROR TABLE POINTER
5201  024016  001416                    BEQ     4$              ;EXIT IF 0
5202  024020  005003                    CLR     R3              ;COUNTER & POINTER
5203  024022  012705  000020            MOV     #RPM2,R5        ;PROBLEM REGISTER
5204  024026  020305            1$:     CMP     R3,R5           ;REACHED RPM2 ?
5205  024030  001002                    BNE     2$              ;BR IF NO
5206  024032  005723                    TST     (R3)+           ;INCREMENT THE INDEX
5207  024034  005723                    TST     (R3)+           ;INCREMENT THE INDEX
5208  024036  010446            2$:     MOV     R4,-(SP)        ;FORM RP11 ADDRESS THAT IS
5209  024040  060316                    ADD     R3,(SP)         ;TO BE READ
5210  024042  013622                    MOV     @(SP)+,(R2)+    ;AND READ IT
5211  024044  005723            3$:     TST     (R3)+           ;MOVE TO NEXT REG INDEX
5212  024046  020327  000026            CMP     R3,#SILO        ;DONE?
5213  024052  003765                    BLE     1$              ;BRANCH IF NO
5214  024054  104413            4$:     RESREG                  ;RESTORE R0-R5
5215  024056  000207                    RTS     PC              ;RETURN TO USER
5216
5217                                    ;ROUTINE TO PUT THE REQUEST IN THE PROPER QUEUE
```

```
                                        ;
                                        ;CALL
                                        ;       MOV     #OPBNUM,R2      ;ADDRESS OF PARAMETER BLOCK
                                        ;       JSR     R0,LOQUE        ;PUT THE REQUEST IN THE PROPER QUEUE
                                        ;       RETURN1                 ;RETURN HERE IF QUEUE IF FULL
                                        ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
                                        ;
024060  010146                  LOQUE:  MOV     R1,-(SP)        ;SAVE R1
024062  032762  000001  000002          BIT     #BIT0,2(R2)     ;IS THIS A GET REGISTER REQUEST ?
024070  001003                          BNE     1$              ;BR IF NOT
024072  012701  021134                  MOV     #QREGS,R1       ;LOAD QUEUE ADDRESS
024076  000445                          BR      7$              ;GO ENTER REQUEST
024100  122762  000011  000002  1$:     CMPB    #SEEK,2(R2)     ;SEEK REQUEST ?
024106  001404                          BEQ     2$              ;BR IF IT IS
024110  122762  000015  000002          CMPB    #HOMSEK,2(R2)   ;HOME SEEK REQUEST ?
024116  001003                          BNE     3$              ;BR IF NOT
024120  012701  021156          2$:     MOV     #QSEEK,R1       ;QUEUE ADDRESS
024124  000432                          BR      7$              ;ENTER REQUEST
024126  122762  000005  000002  3$:     CMPB    #RDSEK,2(R2)    ;READ W/IMPLIED SEEK ?
024134  001424                          BEQ     5$              ;BR IF IT IS
024136  122762  000003  000002          CMPB    #WRTSEK,2(R2)   ;WRITE W/IMPLIED SEEK ?
024144  001420                          BEQ     5$              ;BR IF IT IS
024146  122762  000007  000002          CMPB    #WRTCHK,2(R2)   ;WRITE CHECK ?
024154  001414                          BEQ     5$              ;BR IF WRITE CHECK
024156  122762  000017  000002          CMPB    #READ,2(R2)     ;READ W/O IMPLIED SEEK ?
024164  001410                          BEQ     5$              ;BR IF IT IS
024166  122762  000013  000002          CMPB    #WRITE,2(R2)    ;WRITE W/O IMPLIED SEEK ?
024174  001404                          BEQ     5$              ;BR IF IT IS
024176  012762  120000  000016          MOV     #BIT15!BIT13,16(R2)  ;SET INVALID OPCODE INDICATOR
024204  000405                          BR      8$              ;EXIT WITHOUT MAKING QUEUE ENTRY
024206  012701  021200          5$:     MOV     #QTRANS,R1      ;QUEUE ADDRESS
024212  004037  024226          7$:     JSR     R0,DRVQUE       ;PUT THE REQUEST IN THE QUEUE
024216  000401                          BR      9$              ;QUEUE WAS FULL
024220  005720                  8$:     TST     (R0)+           ;INCREMENT RETURN
024222  012601                  9$:     MOV     (SP)+,R1        ;RESTORE R1
024224  000200                          RTS     R0              ;RETURN

                                ;ROUTINE TO PUT A REQUEST IN QUEUE
                                ;
                                ;CALL
                                ;       MOV     #QNAME,R1       ;QUEUE ADDRESS
                                ;       MOV     #OPB,R2         ;ADDRESS OF PARAMETER BLOCK
                                ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
                                ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
                                ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
                                ;
024226  012746  000010          DRVQUE: MOV     #8.,-(SP)       ;QUEUE SIZE COUNTER
024232  005721                  1$:     TST     (R1)+           ;SEE IF AT END OF ENTRIES
024234  001403                          BEQ     2$              ;BR IF YES
024236  005316                          DEC     (SP)            ;DECREMENT THE COUNTER
024240  003374                          BGT     1$              ;BR IF NOT AT END
024242  000403                          BR      3$              ;AT END
024244  010261  177776          2$:     MOV     R2,-2(R1)       ;PUT REQUEST IN THE QUEUE
024250  005720                          TST     (R0)+           ;INCREMENT RETURN
024252  005726                  3$:     TST     (SP)+           ;CORRECT THE STACK POINTER
024254  000200                          RTS     R0              ;RETURN
```

```
                                ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
                                ;
                                ;CALL
                                ;       MOV     #QNAME,R1       ;QUEUE ADDRESS TO R1
                                ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
                                ;
024256  016111  000002  POPQUE: MOV     2(R1),(R1)      ;SHIFT ENTRY DOWN
024262  001403                  BEQ     1$              ;BR IF AT END
024264  062701  000002          ADD     #2,R1           ;INCREMENT THE POINTER
024270  000772                  BR      POPQUE          ;CONTINUE
024272  000207          1$:     RTS     PC              ;RETURN
                                ;
                                ;;******************************************************************
                                .SBTTL  DATA, CONTROL, & STATUS BLOCKS
                                ;;******************************************************************
                                ;BLOCK LOCATION EQUATE STATEMENTS
                                ;
000001                  $MEX    =       1               ;EXTENDED MEMORY BITS
000002                  $COMND  =       $MEX+1          ;OPERATION CODE
000003                  $MODE   =       $COMND+1        ;MODE & HDR BITS
000004                  $WRDM   =       $MODE+1         ;WORD COUNT (2'S COMP)
000006                  $BUF    =       $WRDM+2         ;BUFFER ADDR
000010                  $CYL    =       $BUF+2          ;CYLINDER ADDRESS
000012                  $SEC    =       $CYL+2          ;SECTOR ADDRESS
000013                  $TRK    =       $SEC+1          ;TRACK ADDRESS OF LAST REG ADDR
000014                  $REG    =       $TRK+1          ;REGISTER STORAGE ADDRESS
000016                  $STATUS =       $REG+2          ;STATUS WORD (SET BY DRIVER)
000020                  $WRDL   =       $STATUS+2       ;WORD COUNT (NOT 2'S COMP)
000022                  $PACK   =       $WRDL+2         ;WRITE DATA PACK INDICATOR
000024                  $PREVO  =       $PACK+2         ;PREVIOUS COMMAND SELECTION CODE
000026                  $PATTC  =       $PREVO+2        ;PATTERN CODE
000030                  $PREVA  =       $PATTC+2        ;PREVIOUS ADDRESS - TRACK, SECTOR, CYLINDER
000034                  $OPERC  =       $PREVA+4        ;OPERATION COUNT
000040                  $POSIT  =       $OPERC+4        ;SEEK COUNT
000044                  $READ   =       $POSIT+4        ;TOTAL BITS READ COUNT
000050                  $TOTAL  =       $READ+4         ;TOTAL ERRORS (ALL TYPES) COUNT
000052                  $SOFT   =       $TOTAL+2        ;'SOFT' ERROR COUNT
000054                  $HARD   =       $SOFT+2         ;'HARD' ERROR COUNT
000056                  $SKI    =       $HARD+2         ;'SUSI' ERROR COUNT
000060                  $MISPO  =       $SKI+2          ;POSITIONING ERROR COUNT
000062                  $PASSC  =       $MISPO+2        ;PASS COUNTER
000064                  $FAIR   =       $PASSC+2        ;BUFFER WAIT COUNT
000066                  $RPDS   =       $FAIR+2         ;RP11E REGISTER STORAGE
000070                  $RPER   =       $RPDS+2         ;ERROR REGISTER
000072                  $RPCS   =       $RPER+2         ;CONTROL REGISTER
000074                  $RPWC   =       $RPCS+2         ;WORD COUNT REGISTER
000076                  $RPBA   =       $RPWC+2         ;BUFFER ADDRESS REGISTER
000100                  $RPCA   =       $RPBA+2         ;CYLINDER ADDRESS REGISTER
000102                  $RPDA   =       $RPCA+2         ;SECTOR/ADDRESS REGISTER
000104                  $RPMI   =       $RPDA+2         ;MAINTENANCE REGISTER #1
000106                  $SUCA   =       $RPMI+2         ;SELECTED UNIT CYLINDER ADDR REGISTER
000110                  $SILO   =       $SUCA+2         ;SILO REGISTER
```

```
5330
5331
5332                                    ;BLOCK FOR DRIVE 0
5333
5334  024274    000     000     BLK0:   .BYTE   0,0             ;DRIVE NUMBER
5335  024276  000000                    .WORD   0
5336  024300  000000                    .WORD   0
5337  024302  000000                    .WORD   0
5338  024304  000000                    .WORD   0
5339  024306  000000                    .WORD   0
5340  024310  024362                    .WORD   .+52
5341  024312  000000                    .WORD   0
5342  024314  000000                    .WORD   0
5343  024316  000000                    .WORD   0
5344  024320  000000                    .WORD   0
5345  024322  000000                    .WORD   0
5346  024324  000000                    .WORD   0
5347  024326  000000                    .WORD   0
5348  024330  000000                    .WORD   0
5349  024332  000000                    .WORD   0
5350  024334  000000                    .WORD   0
5351  024336  000000                    .WORD   0
5352  024340  000000                    .WORD   0
5353  024342  000000                    .WORD   0
5354  024344  000000                    .WORD   0
5355  024346  000000                    .WORD   0
5356  024350  000000                    .WORD   0
5357  024352  000000                    .WORD   0
5358  024354  000000                    .WORD   0
5359  024356  000000                    .WORD   0
5360  024360  000000                    .WORD   0
5361  024362  000000                    .WORD   0
5362  024364  000000                    .WORD   0
5363  024366  000000                    .WORD   0
5364  024370  000000                    .WORD   0
5365  024372  000000                    .WORD   0
5366  024374  000000                    .WORD   0
5367  024376  000000                    .WORD   0
5368  024400  000000                    .WORD   0
5369  024402  000000                    .WORD   0
5370  024404  000000                    .WORD   0
5371
5372                                    ;BLOCK FOR DRIVE 1
5373
5374  024406    001     000     BLK1:   .BYTE   1,0             ;DRIVE NUMBER
5375  024410  000000                    .WORD   0
5376  024412  000000                    .WORD   0
5377  024414  000000                    .WORD   0
5378  024416  000000                    .WORD   0
5379  024420  000000                    .WORD   0
5380  024422  024474                    .WORD   .+52
5381  024424  000000                    .WORD   0
5382  024426  000000                    .WORD   0
5383  024430  000000                    .WORD   0
5384  024432  000000                    .WORD   0
5385  024434  000000                    .WORD   0
```

```
5386  024436  000000                          .WORD   0
5387  024440  000000                          .WORD   0
5388  024442  000000                          .WORD   0
5389  024444  000000                          .WORD   0
5390  024446  000000                          .WORD   0
5391  024450  000000                          .WORD   0
5392  024452  000000                          .WORD   0
5393  024454  000000                          .WORD   0
5394  024456  000000                          .WORD   0
5395  024460  000000                          .WORD   0
5396  024462  000000                          .WORD   0
5397  024464  000000                          .WORD   0
5398  024466  000000                          .WORD   0
5399  024470  000000                          .WORD   0
5400  024472  000000                          .WORD   0
5401  024474  000000                          .WORD   0
5402  024476  000000                          .WORD   0
5403  024500  000000                          .WORD   0
5404  024502  000000                          .WORD   0
5405  024504  000000                          .WORD   0
5406  024506  000000                          .WORD   0
5407  024510  000000                          .WORD   0
5408  024512  000000                          .WORD   0
5409  024514  000000                          .WORD   0
5410  024516  000000                          .WORD   0
5411
5412                                  ;BLOCK FOR DRIVE 2
5413
5414  024520      002     000         BLK2:   .BYTE   2,0             ;DRIVE NUMBER
5415  024522  000000                          .WORD   0
5416  024524  000000                          .WORD   0
5417  024526  000000                          .WORD   0
5418  024530  000000                          .WORD   0
5419  024532  000000                          .WORD   0
5420  024534  024606                          .WORD   .+52
5421  024536  000000                          .WORD   0
5422  024540  000000                          .WORD   0
5423  024542  000000                          .WORD   0
5424  024544  000000                          .WORD   0
5425  024546  000000                          .WORD   0
5426  024550  000000                          .WORD   0
5427  024552  000000                          .WORD   0
5428  024554  000000                          .WORD   0
5429  024556  000000                          .WORD   0
5430  024560  000000                          .WORD   0
5431  024562  000000                          .WORD   0
5432  024564  000000                          .WORD   0
5433  024566  000000                          .WORD   0
5434  024570  000000                          .WORD   0
5435  024572  000000                          .WORD   0
5436  024574  000000                          .WORD   0
5437  024576  000000                          .WORD   0
5438  024600  000000                          .WORD   0
5439  024602  000000                          .WORD   0
5440  024604  000000                          .WORD   0
5441  024606  000000                          .WORD   0
```

```
5442  024610  000000                          .WORD   0
5443  024612  000000                          .WORD   0
5444  024614  000000                          .WORD   0
5445  024616  000000                          .WORD   0
5446  024620  000000                          .WORD   0
5447  024622  000000                          .WORD   0
5448  024624  000000                          .WORD   0
5449  024626  000000                          .WORD   0
5450  024630  000000                          .WORD   0
5451
5452                                  ;BLOCK FOR DRIVE 3
5453
5454  024632     003     000        BLK3:    .BYTE   3,0             ;DRIVE NUMBER
5455  024634  000000                          .WORD   0
5456  024636  000000                          .WORD   0
5457  024640  000000                          .WORD   0
5458  024642  000000                          .WORD   0
5459  024644  000000                          .WORD   0
5460  024646  024720                          .WORD   .+52
5461  024650  000000                          .WORD   0
5462  024652  000000                          .WORD   0
5463  024654  000000                          .WORD   0
5464  024656  000000                          .WORD   0
5465  024660  000000                          .WORD   0
5466  024662  000000                          .WORD   0
5467  024664  000000                          .WORD   0
5468  024666  000000                          .WORD   0
5469  024670  000000                          .WORD   0
5470  024672  000000                          .WORD   0
5471  024674  000000                          .WORD   0
5472  024676  000000                          .WORD   0
5473  024700  000000                          .WORD   0
5474  024702  000000                          .WORD   0
5475  024704  000000                          .WORD   0
5476  024706  000000                          .WORD   0
5477  024710  000000                          .WORD   0
5478  024712  000000                          .WORD   0
5479  024714  000000                          .WORD   0
5480  024716  000000                          .WORD   0
5481  024720  000000                          .WORD   0
5482  024722  000000                          .WORD   0
5483  024724  000000                          .WORD   0
5484  024726  000000                          .WORD   0
5485  024730  000000                          .WORD   0
5486  024732  000000                          .WORD   0
5487  024734  000000                          .WORD   0
5488  024736  000000                          .WORD   0
5489  024740  000000                          .WORD   0
5490  024742  000000                          .WORD   0
5491
5492                                  ;BLOCK FOR DRIVE 4
5493
5494  024744     004     000        BLK4:    .BYTE   4,0             ;DRIVE NUMBER
5495  024746  000000                          .WORD   0
5496  024750  000000                          .WORD   0
5497  024752  000000                          .WORD   0
```

```
5498   024754  000000                          .WORD   0
5499   024756  000000                          .WORD   0
5500   024760  025032                          .WORD   .+52
5501   024762  000000                          .WORD   0
5502   024764  000000                          .WORD   0
5503   024766  000000                          .WORD   0
5504   024770  000000                          .WORD   0
5505   024772  000000                          .WORD   0
5506   024774  000000                          .WORD   0
5507   024776  000000                          .WORD   0
5508   025000  000000                          .WORD   0
5509   025002  000000                          .WORD   0
5510   025004  000000                          .WORD   0
5511   025006  000000                          .WORD   0
5512   025010  000000                          .WORD   0
5513   025012  000000                          .WORD   0
5514   025014  000000                          .WORD   0
5515   025016  000000                          .WORD   0
5516   025020  000000                          .WORD   0
5517   025022  000000                          .WORD   0
5518   025024  000000                          .WORD   0
5519   025026  000000                          .WORD   0
5520   025030  000000                          .WORD   0
5521   025032  000000                          .WORD   0
5522   025034  000000                          .WORD   0
5523   025036  000000                          .WORD   0
5524   025040  000000                          .WORD   0
5525   025042  000000                          .WORD   0
5526   025044  000000                          .WORD   0
5527   025046  000000                          .WORD   0
5528   025050  000000                          .WORD   0
5529   025052  000000                          .WORD   0
5530   025054  000000                          .WORD   0
5531
5532                                   ;BLOCK FOR DRIVE 5
5533
5534   025056     005  000            BLK5:   .BYTE   5,0             ;DRIVE NUMBER
5535   025060  000000                          .WORD   0
5536   025062  000000                          .WORD   0
5537   025064  000000                          .WORD   0
5538   025066  000000                          .WORD   0
5539   025070  000000                          .WORD   0
5540   025072  025144                          .WORD   .+52
5541   025074  000000                          .WORD   0
5542   025076  000000                          .WORD   0
5543   025100  000000                          .WORD   0
5544   025102  000000                          .WORD   0
5545   025104  000000                          .WORD   0
5546   025106  000000                          .WORD   0
5547   025110  000000                          .WORD   0
5548   025112  000000                          .WORD   0
5549   025114  000000                          .WORD   0
5550   025116  000000                          .WORD   0
5551   025120  000000                          .WORD   0
5552   025122  000000                          .WORD   0
5553   025124  000000                          .WORD   0
```

```
5554  025126  000000                          .WORD   0
5555  025130  000000                          .WORD   0
5556  025132  000000                          .WORD   0
5557  025134  000000                          .WORD   0
5558  025136  000000                          .WORD   0
5559  025140  000000                          .WORD   0
5560  025142  000000                          .WORD   0
5561  025144  000000                          .WORD   0
5562  025146  000000                          .WORD   0
5563  025150  000000                          .WORD   0
5564  025152  000000                          .WORD   0
5565  025154  000000                          .WORD   0
5566  025156  000000                          .WORD   0
5567  025160  000000                          .WORD   0
5568  025162  000000                          .WORD   0
5569  025164  000000                          .WORD   0
5570  025166  000000                          .WORD   0
5571
5572                                  ;BLOCK FOR DRIVE 6
5573
5574  025170    006     000         BLK6:     .BYTE   6,0             ;DRIVE NUMBER
5575  025172  000000                          .WORD   0
5576  025174  000000                          .WORD   0
5577  025176  000000                          .WORD   0
5578  025200  000000                          .WORD   0
5579  025202  000000                          .WORD   0
5580  025204  025256                          .WORD   .+52
5581  025206  000000                          .WORD   0
5582  025210  000000                          .WORD   0
5583  025212  000000                          .WORD   0
5584  025214  000000                          .WORD   0
5585  025216  000000                          .WORD   0
5586  025220  000000                          .WORD   0
5587  025222  000000                          .WORD   0
5588  025224  000000                          .WORD   0
5589  025226  000000                          .WORD   0
5590  025230  000000                          .WORD   0
5591  025232  000000                          .WORD   0
5592  025234  000000                          .WORD   0
5593  025236  000000                          .WORD   0
5594  025240  000000                          .WORD   0
5595  025242  000000                          .WORD   0
5596  025244  000000                          .WORD   0
5597  025246  000000                          .WORD   0
5598  025250  000000                          .WORD   0
5599  025252  000000                          .WORD   0
5600  025254  000000                          .WORD   0
5601  025256  000000                          .WORD   0
5602  025260  000000                          .WORD   0
5603  025262  000000                          .WORD   0
5604  025264  000000                          .WORD   0
5605  025266  000000                          .WORD   0
5606  025270  000000                          .WORD   0
5607  025272  000000                          .WORD   0
5608  025274  000000                          .WORD   0
5609  025276  000000                          .WORD   0
```

```
5610  025300  000000                          .WORD   0
5611
5612                                  ;BLOCK FOR DRIVE 7
5613
5614  025302    007   000           BLK7:   .BYTE   7,0             ;DRIVE NUMBER
5615  025304  000000                        .WORD   0
5616  025306  000000                        .WORD   0
5617  025310  000000                        .WORD   0
5618  025312  000000                        .WORD   0
5619  025314  000000                        .WORD   0
5620  025316  025370                        .WORD   .+52
5621  025320  000000                        .WORD   0
5622  025322  000000                        .WORD   0
5623  025324  000000                        .WORD   0
5624  025326  000000                        .WORD   0
5625  025330  000000                        .WORD   0
5626  025332  000000                        .WORD   0
5627  025334  000000                        .WORD   0
5628  025336  000000                        .WORD   0
5629  025340  000000                        .WORD   0
5630  025342  000000                        .WORD   0
5631  025344  000000                        .WORD   0
5632  025346  000000                        .WORD   0
5633  025350  000000                        .WORD   0
5634  025352  000000                        .WORD   0
5635  025354  000000                        .WORD   0
5636  025356  000000                        .WORD   0
5637  025360  000000                        .WORD   0
5638  025362  000000                        .WORD   0
5639  025364  000000                        .WORD   0
5640  025366  000000                        .WORD   0
5641  025370  000000                        .WORD   0
5642  025372  000000                        .WORD   0
5643  025374  000000                        .WORD   0
5644  025376  000000                        .WORD   0
5645  025400  000000                        .WORD   0
5646  025402  000000                        .WORD   0
5647  025404  000000                        .WORD   0
5648  025406  000000                        .WORD   0
5649  025410  000000                        .WORD   0
5650  025412  000000                        .WORD   0
5651
5652                                  ;GENERAL PURPOSE DPB - USED BY 'READHD' & 'RECALT'
5653
5654  025414  000000  000000  177775  GENDPB: .WORD   0,0,-3,CYLDER
5655  025422  033636
5656  025424  000000  000000  025434          .WORD   0,0,GENREG,0
5657  025432  000000
5658  025434  000012                  GENREG: .BLKW   12              ;REGISTER STORAGE IF ERROR
5659
5660                                  ;;*****************************************************************
5661
5662                                  .SBTTL  TABLES, CONSTANTS, AND VARIABLE LOCATIONS
5663
5664                                  ;;*****************************************************************
5665
```

```
5666  025460  000000                  ASNLST: .WORD   0               ;BIT SET FOR EACH ASSIGNED DRIVE
5667
5668  025462  000000  000000  000000  ORDERQ: .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF DRIVES PERFORMING COMMANDS
5669  025470  000000  000000  000000
5670  025476  000000  000000  000000
5671
5672  025504  000000  000000  000000  DUNIT:  .WORD   0,0,0,0,0,0,0,0,0  ;ADDRESSES OF DRIVES TO BE DEASSIGNED
5673  025512  000000  000000  000000
5674  025520  000000  000000  000000
5675
5676  025526  000000  000000  000000  NEWUNT: .WORD   0,0,0,0,0,0,0,0,0  ;ADDRESSES OF NEWLY ASSIGNED DRIVES
5677  025534  000000  000000  000000
5678  025542  000000  000000  000000
5679
5680  025550  000000  000000  000000  AVAIL:  .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF DRIVES WAITING FOR PARAMETERS
5681  025556  000000  000000  000000
5682  025564  000000  000000  000000
5683
5684  025572  000000  000000  000000  WAIT:   .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF DRIVES WAITING FOR BUFFERS
5685  025600  000000  000000  000000
5686  025606  000000  000000  000000
5687
5688  025614  000000                  BUFTBL: .WORD   0               ;BUFFER ALLOCATION TABLE ENTRY COUNT
5689  025616  000000  000000                  .WORD   0,0
5690  025622  000000  000000                  .WORD   0,0
5691  025626  000000  000000                  .WORD   0,0
5692  025632  000000  000000                  .WORD   0,0
5693  025636  000000  000000                  .WORD   0,0
5694  025642  000000  000000                  .WORD   0,0
5695  025646  000000  000000                  .WORD   0,0
5696  025652  000000  000000                  .WORD   0,0
5697  025656  000000  000000                  .WORD   0,0
5698  025662  000000  000000                  .WORD   0,0
5699  025666  000000  000000                  .WORD   0,0
5700  025672  000000  000000                  .WORD   0,0
5701  025676  000000  000000                  .WORD   0,0
5702  025702  000000  000000                  .WORD   0,0
5703  025706  000000  000000                  .WORD   0,0
5704  025712  000000  000000                  .WORD   0,0
5705  025716  000000  000000                  .WORD   0,0
5706  025722  000000  000000                  .WORD   0,0
5707  025726  000000  000000                  .WORD   0,0
5708  025732  000000  000000                  .WORD   0,0
5709
5710  025736                          BLKADR:                 ;ADDRESS OF EACH UNIT BLOCK START
5711  025736  024274                          .WORD   BLK0
5712  025740  024406                          .WORD   BLK1
5713  025742  024520                          .WORD   BLK2
5714  025744  024632                          .WORD   BLK3
5715  025746  024744                          .WORD   BLK4
5716  025750  025056                          .WORD   BLK5
5717  025752  025170                          .WORD   BLK6
5718  025754  025302                          .WORD   BLK7
5719
5720  025756      003                 COMTBL: .BYTE   WRTSEK          ;WRITE WITH IMPLIED SEEK (CODE = 0)
5721  025757      013                         .BYTE   WRITE           ;WRITE WITHOUT IMPLIED SEEK (CODE = 1)
```

```
5722  025760  007                     .BYTE   WRTCHK          ;WRITE CHECK (CODE = 2)
5723  025761  005                     .BYTE   RDSEK           ;READ WITH IMPLIED SEEK (CODE = 3)
5724  025762  017                     .BYTE   READ            ;READ WITHOUT IMPLIED SEEK (CODE = 4)
5725
5726
5727          025764                  .EVEN
5728
5729                                ;;*****************************************************************
5730
5731                                .SBTTL  DATA PATTERNS
5732
5733                                ;;*****************************************************************
5734
5735  025764  000000                STNDAT: .WORD   0               ;STANDARD DATA PATTERN POINTER TABLE
5736  025766  026070                        .WORD   DATA1
5737  025770  026130                        .WORD   DATA1+40
5738  025772  026170                        .WORD   DATA1+100
5739  025774  026230                        .WORD   DATA1+140
5740  025776  026270                        .WORD   DATA1+200
5741  026000  026330                        .WORD   DATA1+240
5742  026002  026370                        .WORD   DATA1+300
5743  026004  026430                        .WORD   DATA1+340
5744  026006  026470                        .WORD   DATA1+400
5745  026010  026530                        .WORD   DATA1+440
5746  026012  026570                        .WORD   DATA1+500
5747  026014  026630                        .WORD   DATA1+540
5748  026016  026670                        .WORD   DATA1+600
5749  026020  026730                        .WORD   DATA1+640
5750  026022  026770                        .WORD   DATA1+700
5751  026024  026030                        .WORD   DATA0           ;ZEROES
5752  026026  026732                        .WORD   DATA1+642       ;ONES
5753
5754  026030  000000                DATA0:  .WORD   0       ;STANDARD PATTERN 0
5755  026032  000000                        .WORD   0
5756  026034  000000                        .WORD   0
5757  026036  000000                        .WORD   0
5758  026040  000000                        .WORD   0
5759  026042  000000                        .WORD   0
5760  026044  000000                        .WORD   0
5761  026046  000000                        .WORD   0
5762  026050  000000                        .WORD   0
5763  026052  000000                        .WORD   0
5764  026054  000000                        .WORD   0
5765  026056  000000                        .WORD   0
5766  026060  000000                        .WORD   0
5767  026062  000000                        .WORD   0
5768  026064  000000                        .WORD   0
5769  026066  000000                        .WORD   0
5770
5771  026070  000001                DATA1:  .WORD   000001  ;STANDARD PATTERN 1
5772  026072  000003                        .WORD   000003
5773  026074  000007                        .WORD   000007
5774  026076  000017                        .WORD   000017
5775  026100  000037                        .WORD   000037
5776  026102  000077                        .WORD   000077
5777  026104  000177                        .WORD   000177
```

```
5778  026106  000377                .WORD   000377
5779  026110  000777                .WORD   000777
5780  026112  001777                .WORD   001777
5781  026114  003777                .WORD   003777
5782  026116  007777                .WORD   007777
5783  026120  017777                .WORD   017777
5784  026122  037777                .WORD   037777
5785  026124  077777                .WORD   077777
5786  026126  177777                .WORD   177777
5787
5788  026130  177776                .WORD   177776  ;STANDARD PATTERN 2
5789  026132  177774                .WORD   177774
5790  026134  177770                .WORD   177770
5791  026136  177760                .WORD   177760
5792  026140  177740                .WORD   177740
5793  026142  177700                .WORD   177700
5794  026144  177600                .WORD   177600
5795  026146  177400                .WORD   177400
5796  026150  177000                .WORD   177000
5797  026152  176000                .WORD   176000
5798  026154  174000                .WORD   174000
5799  026156  170000                .WORD   170000
5800  026160  160000                .WORD   160000
5801  026162  140000                .WORD   140000
5802  026164  100000                .WORD   100000
5803  026166  000000                .WORD   000000
5804
5805  026170  000000                .WORD   000000  ;STANDARD PATTERN 3
5806  026172  000000                .WORD   000000
5807  026174  000000                .WORD   000000
5808  026176  177777                .WORD   177777
5809  026200  177777                .WORD   177777
5810  026202  177777                .WORD   177777
5811  026204  000000                .WORD   000000
5812  026206  000000                .WORD   000000
5813  026210  177777                .WORD   177777
5814  026212  177777                .WORD   177777
5815  026214  000000                .WORD   000000
5816  026216  177777                .WORD   177777
5817  026220  000000                .WORD   000000
5818  026222  177777                .WORD   177777
5819  026224  000000                .WORD   000000
5820  026226  177777                .WORD   177777
5821
5822  026230  000000                .WORD   000000  ;STANDARD PATTERN 4
5823  026232  010421                .WORD   010421
5824  026234  021042                .WORD   021042
5825  026236  031463                .WORD   031463
5826  026240  042104                .WORD   042104
5827  026242  052525                .WORD   052525
5828  026244  063146                .WORD   063146
5829  026246  073567                .WORD   073567
5830  026250  104210                .WORD   104210
5831  026252  114631                .WORD   114631
5832  026254  125252                .WORD   125252
5833  026256  135673                .WORD   135673
```

```
                        026260  146314          .WORD   146314
                        026262  156735          .WORD   156735
                        026264  167356          .WORD   167356
                        026266  177777          .WORD   177777

                        026270  052525          .WORD   052525  ;STANDARD PATTERN 5
                        026272  052525          .WORD   052525
                        026274  052525          .WORD   052525
                        026276  125252          .WORD   125252
                        026300  125252          .WORD   125252
                        026302  125252          .WORD   125252
                        026304  052525          .WORD   052525
                        026306  052525          .WORD   052525
                        026310  125252          .WORD   125252
                        026312  125252          .WORD   125252
                        026314  052525          .WORD   052525
                        026316  125252          .WORD   125252
                        026320  052525          .WORD   052525
                        026322  125252          .WORD   125252
                        026324  052525          .WORD   052525
                        026326  125252          .WORD   125252

                        026330  007417          .WORD   007417  ;STANDARD PATTERN 6
                        026332  007417          .WORD   007417
                        026334  007417          .WORD   007417
                        026336  170360          .WORD   170360
                        026340  170360          .WORD   170360
                        026342  170360          .WORD   170360
                        026344  007417          .WORD   007417
                        026346  007417          .WORD   007417
                        026350  170360          .WORD   170360
                        026352  170360          .WORD   170360
                        026354  007417          .WORD   007417
                        026356  170360          .WORD   170360
                        026360  007417          .WORD   007412
                        026362  170360          .WORD   170360
                        026364  007417          .WORD   007417
                        026366  170360          .WORD   170360

                        026370  026455          .WORD   026455  ;STANDARD PATTERN 7
                        026372  026455          .WORD   026455
                        026374  026455          .WORD   026455
                        026376  151322          .WORD   151322
                        026400  151322          .WORD   151322
                        026402  151322          .WORD   151322
                        026404  026455          .WORD   026455
                        026406  026455          .WORD   026455
                        026410  151322          .WORD   151322
                        026412  151322          .WORD   151322
                        026414  026455          .WORD   026455
                        026416  151322          .WORD   151322
                        026420  026455          .WORD   026455
                        026422  151322          .WORD   151322
                        026424  026455          .WORD   026455
                        026426  151322          .WORD   151322
```

```
 5890  026430  077577                .WORD   077577  ;STANDARD PATTERN 8 (WORST CASE PATTERN)
 5891  026432  077577                .WORD   077577
 5892  026434  077577                .WORD   077577
 5893  026436  077577                .WORD   077577
 5894  026440  077577                .WORD   077577
 5895  026442  077577                .WORD   077577
 5896  026444  077577                .WORD   077577
 5897  026446  077577                .WORD   077577
 5898  026450  077577                .WORD   077577
 5899  026452  077577                .WORD   077577
 5900  026454  077577                .WORD   077577
 5901  026456  077577                .WORD   077577
 5902  026460  077577                .WORD   077577
 5903  026462  077577                .WORD   077577
 5904  026464  077577                .WORD   077577
 5905  026466  077577                .WORD   077577
 5906
 5907  026470  000001                .WORD   000001  ;STANDARD PATTERN 9
 5908  026472  000002                .WORD   000002
 5909  026474  000004                .WORD   000004
 5910  026476  000010                .WORD   000010
 5911  026500  000020                .WORD   000020
 5912  026502  000040                .WORD   000040
 5913  026504  000100                .WORD   000100
 5914  026506  000200                .WORD   000200
 5915  026510  000400                .WORD   000400
 5916  026512  001000                .WORD   001000
 5917  026514  002000                .WORD   002000
 5918  026516  004000                .WORD   004000
 5919  026520  010000                .WORD   010000
 5920  026522  020000                .WORD   020000
 5921  026524  040000                .WORD   040000
 5922  026526  100000                .WORD   100000
 5923
 5924  026530  177776                .WORD   177776  ;STANDARD PATTERN 10
 5925  026532  177775                .WORD   177775
 5926  026534  177773                .WORD   177773
 5927  026536  177767                .WORD   177767
 5928  026540  177757                .WORD   177757
 5929  026542  177737                .WORD   177737
 5930  026544  177677                .WORD   177677
 5931  026546  177577                .WORD   177577
 5932  026550  177377                .WORD   177377
 5933  026552  176777                .WORD   176777
 5934  026554  175777                .WORD   175777
 5935  026556  173777                .WORD   173777
 5936  026560  167777                .WORD   167777
 5937  026562  157777                .WORD   157777
 5938  026564  137777                .WORD   137777
 5939  026566  077777                .WORD   077777
 5940
 5941  026570  172666                .WORD   172666  ;STANDARD PATTERN 11
 5942  026572  155555                .WORD   155555
 5943  026574  172666                .WORD   172666
 5944  026576  155555                .WORC   155555
 5945  026600  172666                .WORD   172666
```

```
5946  026602  155555                .WORD   155555
5947  026604  172666                .WORD   172666
5948  026606  155555                .WORD   155555
5949  026610  172666                .WORD   172666
5950  026612  155555                .WORD   155555
5951  026614  172666                .WORD   172666
5952  026616  155555                .WORD   155555
5953  026620  172666                .WORD   172666
5954  026622  155555                .WORD   155555
5955  026624  172666                .WORD   172666
5956  026626  155555                .WORD   155555
5957
5958  026630  077777                .WORD   077777  ;STANDARD PATTERN 12
5959  026632  137777                .WORD   137777
5960  026634  157777                .WORD   157777
5961  026636  167777                .WORD   167777
5962  026640  173777                .WORD   173777
5963  026642  175777                .WORD   175777
5964  026644  176777                .WORD   176777
5965  026646  177377                .WORD   177377
5966  026650  177577                .WORD   177577
5967  026652  177677                .WORD   177677
5968  026654  177737                .WORD   177737
5969  026656  177757                .WORD   177757
5970  026660  177767                .WORD   177767
5971  026662  177773                .WORD   177773
5972  026664  177775                .WORD   177775
5973  026666  177776                .WORD   177776
5974
5975  026670  153333                .WORD   153333  ;STANDARD PATTERN 13
5976  026672  066667                .WORD   066667
5977  026674  153333                .WORD   153333
5978  026676  066667                .WORD   066667
5979  026700  153333                .WORD   153333
5980  026702  066667                .WORD   066667
5981  026704  153333                .WORD   153333
5982  026706  066667                .WORD   066667
5983  026710  153333                .WORD   153333
5984  026712  066667                .WORD   066667
5985  026714  153333                .WORD   153333
5986  026716  066667                .WORD   066667
5987  026720  153333                .WORD   153333
5988  026722  066667                .WORD   066667
5989  026724  153333                .WORD   153333
5990  026726  066667                .WORD   066667
5991
5992  026730  000000                .WORD   000000  ;STANDARD PATTERN 14
5993  026732  000000                .WORD   000000
5994  026734  177777                .WORD   177777
5995  026736  177777                .WORD   177777
5996  026740  177777                .WORD   177777
5997  026742  177777                .WORD   177777
5998  026744  177777                .WORD   177777
5999  026746  177777                .WORD   177777
6000  026750  177777                .WORD   177777
6001  026752  177777                .WORD   177777
```

F10


```
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777

 026770  177777                                 .WORD   177777  ;STANDARD PATTERN 15
 026772  177777                                 .WORD   177777
 026774  000000                                 .WORD   000000
 026776  000000                                 .WORD   000000
 027000  000000                                 .WORD   000000
 027002  000000                                 .WORD   000000
 027004  000000                                 .WORD   000000
 027006  000000                                 .WORD   000000
 027010  000000                                 .WORD   000000
 027012  000000                                 .WORD   000000
 027014  000000                                 .WORD   000000
 027016  000000                                 .WORD   000000
 027020  000000                                 .WORD   000000
 027022  000000                                 .WORD   000000
 027024  000000                                 .WORD   000000
 027026  000000                                 .WORD   000000

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

                                        .SBTTL  PRINTER/TELETYPE MESSAGES

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

 027030  050122  030461  042040  EM1:    .ASCIZ  @RP11 DIDN'T RESPOND TO ADDRESSING@
 027036  042111  023516  020124
 027044  042522  050123  047117
 027052  020104  047524  040440
 027060  042104  042522  051523
 027066  047111  000107

 027072  047111  040526  044514  EM2:    .ASCIZ  @INVALID RP11 COMMAND@
 027100  020104  050122  030461
 027106  041440  046517  040515
 027114  042116     000

 027117     125  042116  043105  EM3:    .ASCIZ  @UNDEFINED INTERRUPT FROM THE RP11@
 027124  047111  042105  044440
 027132  052116  051105  052522
 027140  052120  043040  047522
 027146  020115  044124  020105
 027154  050122  030461     000

 027161     103  047101  052047  EM10:   .ASCIZ  @CAN'T MATCH DATA READ WITH A PATTERN@
 027166  046440  052101  044103
 027174  042040  052101  020101
 027202  042522  042101  053440
 027210  052111  020110  020101
 027216  040520  052124  051105
 027224  000116
```

```

027226  040504  040524  041440  EM12:   .ASCIZ  @DATA COMPARE ERRORS@
027234  046517  040520  042522
027242  042440  051122  051117
027250  000123

027252  042447  051122  020047  EM15:   .ASCIZ  @'ERR' NOT SET BUT 'RPER' CONTENTS NOT ZERO@
027260  047516  020124  042523
027266  020124  052502  020124
027274  051047  042520  023522
027302  041440  047117  042524
027310  052116  020123  047516
027316  020124  042532  047522
027324     000

027325     127  051117  020104  EM16:   .ASCIZ  @WORD COUNT NOT ZERO OR BUFFER ADDRESS WRONG@
027332  047503  047125  020124
027340  047516  020124  042532
027346  047522  047440  020122
027354  052502  043106  051105
027362  040440  042104  042522
027370  051523  053440  047522
027376  043516     000

027401     123  043117  053524  EM17:   .ASCIZ  @SOFTWARE TIMEOUT DURING POSITIONING@
027406  051101  020105  044524
027414  042515  052517  020124
027422  052504  044522  043516
027430  050040  051517  052111
027436  047511  044516  043516
027444     000

027445     123  043117  053524  EM20:   .ASCIZ  @SOFTWARE TIMEOUT DURING I/O@
027452  051101  020105  044524
027460  042515  052517  020124
027466  052504  044522  043516
027474  044440  047457     000

027501     104  044522  042526  EM21:   .ASCIZ  @DRIVE OFFLINE@
027506  047440  043106  044514
027514  042516     000

027517     104  044522  042526  EM22:   .ASCIZ  @DRIVE UNSAFE@
027524  052440  051516  043101
027532  000105

027534  042523  045505  044440  EM23:   .ASCIZ  @SEEK INCOMPLETE@
027542  041516  046517  046120
027550  052105  000105

027554  047125  041111  051525  EM24:   .ASCIZ  @UNIBUS TRANSFER ERROR@
027562  052040  040522  051516
027570  042506  020122  051105
027576  047522  000122

027602  047503  052116  047522  EM25:   .ASCIZ  @CONTROLLER ERROR@
```

```
 6114  027610  046114  051105  042440
 6115  027616  051122  051117     000
 6116
 6117  027623     116  047117  042455  EM26:   .ASCIZ  @NON-EXISTENT DISK ADDRESS@
 6118  027630  044530  051123  047105
 6119  027636  020124  044504  045523
 6120  027644  040440  042104  042522
 6121  027652  051523     000
 6122
 6123  027655     104  052101  020101  EM27:   .ASCIZ  @DATA ERROR@
 6124  027662  051105  047522  000122
 6125
 6126  027670  051127  052111  020105  EM32:   .ASCIZ  @WRITE CHECK ERROR (NO DATA ERROR)@
 6127  027676  044103  041505  020113
 6128  027704  051105  047522  020122
 6129  027712  047050  020117  040504
 6130  027720  040524  042440  051122
 6131  027726  051117  000051
 6132
 6133  027732  051127  052111  020105  EM33:   .ASCIZ  @WRITE CHECK ERROR (DATA ERROR BITS SET)@
 6134  027740  044103  041505  020113
 6135  027746  051105  047522  020122
 6136  027754  042050  052101  020101
 6137  027762  051105  047522  020122
 6138  027770  044502  051524  051440
 6139  027776  052105  000051
 6140
 6141  030002  042510  042101  051105  EM34:   .ASCIZ  @HEADER NOT FOUND@
 6142  030010  047040  052117  043040
 6143  030016  052517  042116     000
 6144
 6145  030023     105  051117  040515  EM35:   .ASCIZ  @FORMAT ERROR WHILE SEARCHING FOR INITIAL SECTOR@
 6146  030030  020124  051105  047522
 6147  030036  020122  044127  046111
 6148  030044  020105  042523  051101
 6149  030052  044103  047111  020107
 6150  030060  047506  020122  047111
 6151  030066  052111  040511  020114
 6152  030074  042523  052103  051117
 6153  030102     000
 6154
 6155  030103     110  040505  042504  EM36:   .ASCIZ  @HEADER NOT FOUND, CYLINDER IN HEADER DOES NOT COMPARE@
 6156  030110  020122  047516  020124
 6157  030116  047506  047125  026104
 6158  030124  041440  046131  047111
 6159  030132  042504  020122  047111
 6160  030140  044040  040505  042504
 6161  030146  020122  047504  051505
 6162  030154  047040  052117  041440
 6163  030162  046517  040520  042522
 6164  030170     000
 6165
 6166  030171     104  043111  042506  EM37:   .ASCIZ  @DIFFERENT ERROR DURING RETRY@
 6167  030176  042522  052116  042440
 6168  030204  051122  051117  042040
 6169  030212  051125  047111  020107
```

```
 6170  030220  042522  051124  000131
 6171
 6172  030226  042447  051122  020047  EM43:   .ASCIZ  @'ERR' SET BUT NOT ERROR BITS SET@
 6173  030234  042523  020124  052502
 6174  030242  020124  047516  020124
 6175  030250  051105  047522  020122
 6176  030256  044502  051524  051440
 6177  030264  052105     000
 6178
 6179  030267     127  044522  042524  EM44:   .ASCIZ  @WRITE PROTECTION VIOLATION@
 6180  030274  050040  047522  042524
 6181  030302  052103  047511  020116
 6182  030310  044526  046117  052101
 6183  030316  047511  000116
 6184
 6185  030322  047506  046522  052101  EM45:   .ASCIZ  @FORMAT ERROR DURING I/O@
 6186  030330  042440  051122  051117
 6187  030336  042040  051125  047111
 6188  030344  020107  027511  000117
 6189
 6190  030352  050122  042101  000122  DH1:    .ASCIZ  @RPADR@
 6191  030360  051104  020126  020043  DH2:    .ASCIZ  @DRV #   RPCS@
 6192  030366  020040  050122  051503
 6193  030374     000
 6194  030375     122  042120  020123  DH3:    .ASCIZ  @RPDS    RPCS@
 6195  030402  020040  051040  041520
 6196  030410  000123
 6197  030412  020040  020040  020040  DH10:   .ASCII  @       BUFFER@<15><12>
 6198  030420  020040  052502  043106
 6199  030426  051105  005015
 6200  030432  051104  020126  020043          .ASCIZ  @DRV #   START   CYL     TRK     SEC@
 6201  030440  020040  052123  051101
 6202  030446  020124  020040  054503
 6203  030454  020114  020040  020040
 6204  030462  051124  020113  020040
 6205  030470  020040  042523  000103
 6206  030476  042101  051104  020040  DH10A:  .ASCIZ  @ADDR    CONTENTS@
 6207  030504  020040  047503  052116
 6208  030512  047105  051524     000
 6209  030517     040  020040  020040  DH12:   .ASCII  @       GOOD    BAD@<15><12>
 6210  030524  020040  043440  047517
 6211  030532  020104  020040  041040
 6212  030540  042101  005015
 6213  030544  042101  051104  020040          .ASCIZ  @ADDR    DATA    DATA@
 6214  030552  020040  040504  040524
 6215  030560  020040  020040  040504
 6216  030566  040524     000
 6217  030571     124  052117  046101  DH14:   .ASCIZ  @TOTAL COMPARE ERRORS:@
 6218  030576  041440  046517  040520
 6219  030604  042522  042440  051122
 6220  030612  051117  035123     000
 6221  030617     104  053122  021440  DH15:   .ASCIZ  @DRV #   RPDS    RPER    RPCS@
 6222  030624  020040  051040  042120
 6223  030632  020123  020040  051040
 6224  030640  042520  020122  020040
 6225  030646  051040  041520  000123
```

```
6226  030654  020040  020040  020040  DH16:   .ASCII  @       EXPTD   ACTUAL@<15><12>
6227  030662  020040  054105  052120
6228  030670  020104  020040  041501
6229  030676  052524  046101  005015
6230  030704  051104  020126  020043          .ASCIZ  @DRV #  RPBA    RPBA    RPWC    RPDS    RPER    RPCS    @
6231  030712  020040  050122  040502
6232  030720  020040  020040  050122
6233  030726  040502  020040  020040
6234  030734  050122  041527  020040
6235  030742  020040  050122  051504
6236  030750  020040  020040  050122
6237  030756  051105  020040  020040
6238  030764  050122  051503  020040
6239  030772  020040      000
6240  030775      104 053122  021440  DH17:   .ASCIZ  @DRV #  RPDS    RPER    RPCS    RPWC    RPBA@
6241  031002  020040  051040  042120
6242  031010  020123  020040  051040
6243  031016  042520  020122  020040
6244  031024  051040  041520  020123
6245  031032  020040  051040  053520
6246  031040  020103  020040  051040
6247  031046  041120  000101
6248  031052  050122  040503  020040  DH17A:  .ASCIZ  @RPCA    RPDA    RPM1    SUCA    SILO@
6249  031060  020740  050122  040504
6250  031066  020040  020040  050122
6251  031074  030515  020040  020040
6252  031102  052523  040503  020040
6253  031110  020040  044523  047514
6254  031116      000
6255  031117      040 020040  020040  DH23:   .ASCII  @       PREV    DEST@<15><12>
6256  031124  020040  050040  042522
6257  031132  020126  020040  042040
6258  031140  051505  006524      012
6259  031145      104 053122  021440          .ASCIZ  @DRV #  CYL     CYL     RPDS    RPER    RPCS@
6260  031152  020040  041440  046131
6261  031160  020040  020040  041440
6262  031166  046131  020040  020040
6263  031174  051040  042120  020123
6264  031202  020040  051040  042520
6265  031210  020122  020040  051040
6266  031216  041520  000123
6267  031222  050122  040503  020040  DH23A:  .ASCIZ  @RPCA    SUCA@
6268  031230  020040  052523  040503
6269  031236      000
6270  031237      104 053122  021440  DH24:   .ASCIZ  @DRV #  RPDS    RPER    RPCS    RPBA@
6271  031244  020040  051040  042120
6272  031252  020123  020040  051040
6273  031260  042520  020122  020040
6274  031266  051040  041520  020123
6275  031274  020040  051040  041120
6276  031302  000101
6277  031304  051104  020126  020043  DH25:   .ASCIZ  @DRV #  RPDS    RPER    RPCS@
6278  031312  020040  050122  051504
6279  031320  020040  020040  050122
6280  031326  051105  020040  020040
6281  031334  050122  051503      000
```

```
 6282  031341      104  053122  021440  DH26:   .ASCIZ  @DRV #   RPDS    RPER    RPCS    RPCA    RPDA@
 6283  031346  020040  051040  042120
 6284  031354  020123  020040  051040
 6285  031362  042520  020122  020040
 6286  031370  051040  041520  020123
 6287  031376  020040  051040  041520
 6288  031404  020101  020040  051040
 6289  031412  042120  000101
 6290  031416  051104  020126  020043  DH27:   .ASCIZ  @DRV #   CYL     TRK     SEC     RPDS    RPER    RPCS@
 6291  031424  020040  054503  020114
 6292  031432  020040  020040  051124
 6293  031440  020113  020040  020040
 6294  031446  042523  020103  020040
 6295  031454  020040  050122  051504
 6296  031462  020040  020040  050122
 6297  031470  051105  020040  020040
 6298  031476  050122  051503      000
 6299  031503      122  053520  020103  DH27A:  .ASCIZ  @RPWC    RPBA    RPCA    RPDA    RPM1    SUCA    SILO@
 6300  031510  020040  051040  041120
 6301  031516  020101  020040  051040
 6302  031524  041520  020101  020040
 6303  031532  051040  042120  020101
 6304  031540  020040  051040  046520
 6305  031546  020061  020040  051440
 6306  031554  041525  020101  020040
 6307  031562  051440  046111  000117
 6308  031570  047523  052106  042440  DH30:   .ASCIZ  @SOFT ERROR@
 6309  031576  051122  051117      000
 6310  031603      110  051101  020104  DH31:   .ASCIZ  @HARD ERROR@
 6311  031610  051105  047522  000122
 6312  031616  051104  020126  020043  DH34:   .ASCIZ  @DRV # PREV CYL  RPDS    RPER    RPCS    RPDA@
 6313  031624  051120  053105  041440
 6314  031632  046131  020040  050122
 6315  031640  051504  020040  020040
 6316  031646  050122  051105  020040
 6317  031654  020040  050122  051503
 6318  031662  020040  020040  050122
 6319  031670  040504      000
 6320  031673      122  041520  020101  DH34A:  .ASCIZ  @RPCA    SUCA    HEADER (AS READ INTO MEM)@
 6321  031700  020040  051440  041525
 6322  031706  020101  020040  044040
 6323  031714  040505  042504  020122
 6324  031722  040450  020123  042522
 6325  031730  042101  044440  052116
 6326  031736  020117  042515  024515
 6327  031744      000
 6328  031745      122  052105  054522  DH41:   .ASCIZ  @RETRY SUCESSFUL@
 6329  031752  051440  041525  051505
 6330  031760  043123  046125      000
 6331  031765      122  052105  054522  DH42:   .ASCIZ  @RETRY UNSUCESSFUL@
 6332  031772  052440  051516  041525
 6333  032000  051505  043123  046125
 6334  032006      000
 6335
 6336          032010                  .EVEN
 6337
```

```
6338  032010  001216                    DT1:    .WORD   RPADR
6339  032012  001264  001166            DT2:    .WORD   DRIVE,$REGO+RPCS
6340  032016  021104  021106            DT3:    .WORD   RPERRS,RPERRS+2
6341  032022  001122  001126            DT11:   .WORD   $BDADR,$BDDAT
6342  032026  001264  013164  013156    DT10:   .WORD   DRIVE,CMBUF,CMCYL,CMTRK,CMSEC
6343  032034  013162  013160
6344  032040  001122  001124  001126    DT13:   .WORD   $BDADR,$GDDAT,$BDDAT
6345  032046  013150                    DT14:   .WORD   ERCTR
6346  032050  001264  001162  001164    DT15:   .WORD   DRIVE,$REGO+RPDS,$REGO+RPER,$REGO+RPCS
6347  032056  001166
6348  032060  001264  001124  001172    DT16:   .WORD   DRIVE,$GDDAT,$REGO+RPBA,$REGO+RPWC,$REGO+RPDS,$REGO+RPER
6349  032066  001170  001162  001164
6350  032074  001166                            .WORD   $REGO+RPCS
6351  032076  001264  001162  001164    DT17:   .WORD   DRIVE,$REGO+RPDS,$REGO+RPER,$REGO+RPCS,$REGO+RPWC,$REGO+RPBA
6352  032104  001166  001170  001172
6353  032112  001174  001176  001200    DT17A:  .WORD   $REGO+RPCA,$REGO+RPDA,$REGO+RPM1,$REGO+20,$REGO+22
6354  032120  001202  001204
6355  032124  001264  001124  001126    DT23:   .WORD   DRIVE,$GDDAT,$BDDAT,$REGO+RPDS,$REGO+RPER,$REGO+RPCS
6356  032132  001162  001164  001166
6357  032140  001174  001202            DT23A:  .WORD   $REGO+RPCA,$REGO+20
6358  032144  001264  001162  001164    DT24:   .WORD   DRIVE,$REGO+RPDS,$REGO+RPER,$REGO+RPCS,$REGO+RPBA
6359  032152  001166  001172
6360  032156  001264  001162  001164    DT25:   .WORD   DRIVE,$REGO+RPDS,$REGO+RPER,$REGO+RPCS
6361  032164  001166
6362  032166  001264  001162  001164    DT26:   .WORD   DRIVE,$REGO+RPDS,$REGO+RPER,$REGO+RPCS,$REGO+RPCA,$REGO+RPDA
6363  032174  001166  001174  001176
6364  032202  001264  001266  001270    DT27:   .WORD   DRIVE,CYL.ER,TRK.ER,SEC.ER,$REGO+RPDS,$REGO+RPER,$REGO+RPCS
6365  032210  001272  001162  001164
6366  032216  001166
6367  032220  001170  001172  001174    DT27A:  .WORD   $REGO+RPWC,$REGO+RPBA,$REGO+RPCA,$REGO+RPDA,$REGO+RPM1,$REGO+20,$REGO+22
6368  032226  001176  001200  001202
6369  032234  001204
6370  032236  001264  001124  001162    DT34:   .WORD   DRIVE,$GDDAT,$REGO+RPDS,$REGO+RPER,$REGO+RPCS,$REGO+RPDA
6371  032244  001164  001166  001176
6372  032252  001174  001202  033636    DT34A:  .WORD   $REGO+RPCA,$REGO+20,CYLDER,CYLDER+2,CYLDER+4
6373  032260  033640  033642
6374
6375  032264  000001                    DF1:    .WORD   1               ;1 'DH' LINE
6376  032266     001                            .BYTE   1               ;# OF ENTRIES IN 'DH' LINE
6377  032267     000                            .BYTE   0               ;ENTRIES ARE ALL OCTAL
6378
6379
6380  032270  000001                    DF2:    .WORD   1
6381  032272     002                            .BYTE   2               ;2 ENTRIES IN THE 'DH'
6382  032273     000                            .BYTE   0
6383
6384  032274  000002                    DF10:   .WORD   2
6385  032276     005                            .BYTE   5
6386  032277     000                            .BYTE   0
6387  032300  030476                            .WORD   DH10A
6388  032302     000                            .BYTE   0
6389  032303     000                            .BYTE   0
6390
6391  032304  000002                    DF12:   .WORD   2
6392  032306     005                            .BYTE   5
6393  032307     000                            .BYTE   0
```

```
6394  032310  030517                        .WORD   DH12
6395  032312     000                        .BYTE   0
6396  032313     000                        .BYTE   0
6397
6398  032314  000001              DF13:     .WORD   1
6399  032316     003                        .BYTE   3
6400  032317     000                        .BYTE   0
6401
6402  032320  000001              DF14:     .WORD   1
6403  032322     001                        .BYTE   1
6404  032323     001                        .BYTE   1           ;ITEM IS DECIMAL
6405
6406  032324  000001              DF15:     .WORD   1
6407  032326     004                        .BYTE   4
6408  032327     000                        .BYTE   0
6409
6410  032330  000001              DF16:     .WORD   1
6411  032332     007                        .BYTE   7
6412  032333     000                        .BYTE   0
6413
6414  032334  000002              DF17:     .WORD   2           ;2 'DH' LINES
6415  032336     006                        .BYTE   6
6416  032337     000                        .BYTE   0
6417  032340  031052                        .WORD   DH17A
6418  032342     005                        .BYTE   5
6419  032343     000                        .BYTE   0
6420
6421  032344  000002              DF23:     .WORD   2
6422  032346     006                        .BYTE   6
6423  032347     000                        .BYTE   0
6424  032350  031222                        .WORD   DH23A
6425  032352     002                        .BYTE   2
6426  032353     000                        .BYTE   0
6427
6428  032354  000001              DF24:     .WORD   1
6429  032356     005                        .BYTE   5
6430  032357     000                        .BYTE   0
6431
6432  032360  000001              DF26:     .WORD   1
6433  032362     006                        .BYTE   6
6434  032363     000                        .BYTE   0
6435
6436  032364  000002              DF27:     .WORD   2           ;2 'DH' LINES
6437  032366     007                        .BYTE   7
6438  032367     000                        .BYTE   0
6439  032370  031503                        .WORD   DH27A
6440  032372     007                        .BYTE   7
6441  032373     000                        .BYTE   0
6442
6443  032374  000002              DF34:     .WORD   2
6444  032376     006                        .BYTE   6
6445  032377     000                        .BYTE   0
6446  032400  031673                        .WORD   DH34A
6447  032402     005                        .BYTE   5
6448  032403     000                        .BYTE   0
6449
```

```
 6450
 6451  032404  047503  052116  047105  LIN11H: .ASCII  @CONTENTS OF ERROR SECTOR (REPORTED ABOVE)@<15><12>
 6452  032412  051524  047440  020106
 6453  032420  051105  047522  020122
 6454  032426  042523  052103  051117
 6455  032434  024040  042522  047520
 6456  032442  052122  042105  040440
 6457  032450  047502  042526  006451
 6458  032456     012
 6459  032457     101  042104  020122          .ASCIZ  @ADDR    DATA@<15><12>
 6460  032464  020040  042040  052101
 6461  032472  006501  000012
 6462  032476     040                SPACE4: .ASCII  / /
 6463  032477     040                SPACE3: .ASCII  / /
 6464  032500     040                SPACE2: .ASCII  / /
 6465  032501     040     000        SPACE1: .ASCIZ  / /
 6466  032503     122  030120  000062 RP02:   .ASCIZ  /RP02/
 6467  032510  050122  031460     000 RP03:   .ASCIZ  /RP03/
 6468  032515     104  044522  042526 UNTMSG: .ASCIZ  /DRIVE /
 6469  032522  000040
 6470  032524  047440  043106  044514 UNTOFF: .ASCIZ  / OFFLINE/
 6471  032532  042516     000
 6472  032535     040  047117  044514 UNTON:  .ASCIZ  / ONLINE/
 6473  032542  042516     000
 6474  032545     040  047516  020124 UNTNOT: .ASCIZ  / NOT BEING TESTED/
 6475  032552  042502  047111  020107
 6476  032560  042524  052123  042105
 6477  032566     000
 6478  032567     040  046101  042522 UNTASN: .ASCIZ  / ALREADY BEING TESTED/
 6479  032574  042101  020131  042502
 6480  032602  047111  020107  042524
 6481  032610  052123  042105     000
 6482  032615     040  047125  040523 NOTSAF: .ASCIZ  / UNSAFE/
 6483  032622  042506     000
 6484  032625     104  044522  042526 SYSTAT: .ASCIZ  /DRIVE STATUS:/<15><12><12>
 6485  032632  051440  040524  052524
 6486  032640  035123  005015  000012
 6487  032646  051104  053111  020105 STATHD: .ASCII  /DRIVE PERFORMANCE SUMMARY/<15><12>
 6488  032654  042520  043122  051117
 6489  032662  040515  041516  020105
 6490  032670  052523  046515  051101
 6491  032676  006531     012
 6492  032701     104  053122  050040          .ASCII  /DRV PASS ORDERS   SEEKS   WRDS READ/
 6493  032706  051501  020123  051117
 6494  032714  042504  051522  020040
 6495  032722  051440  042505  051513
 6496  032730  020040  053440  042122
 6497  032736  020123  042522  042101
 6498  032744  051440  043117  020124          .ASCIZ  / SOFT HARD SUSI MISP OTHER/<15><12>
 6499  032752  040510  042122  051440
 6500  032760  051525  020111  044515
 6501  032766  050123  047440  044124
 6502  032774  051105  005015     000
 6503  033001     007  043077  052101 DROPNG: .ASCIZ  <07>/?FATAL OR EXCESSIVE ERRORS/
 6504  033006  046101  047440  020122
 6505  033014  054105  042503  051523
```

```
 6506  033022  053111  020105  051105
 6507  033030  047522  051522     000
 6508  033035     105  042116  047440  ENDPAS: .ASCIZ  /END OF PASS/
 6509  033042  020106  040520  051523
 6510  033050     000
 6511  033051     015  042412  042116  ENDTST: .ASCIZ  <15><12>/END OF TEST/
 6512  033056  047440  020106  042524
 6513  033064  052123     000
 6514  033067     040  042504  051501  DEASSG: .ASCIZ  / DEASSIGNED/
 6515  033074  044523  047107  042105
 6516  033102     000
 6517  033103     015  025012  025052  DRNUM:  .ASCIZ  <15><12>/********** DRIVE # /
 6518  033110  025052  025052  025052
 6519  033116  020052  051104  053111
 6520  033124  020105  020043     000
 6521  033131     040  052123  051101  ASGND:  .ASCIZ  / STARTED/<15><12>
 6522  033136  042524  006504  000012
 6523  033144  000077                  QUES:   .ASCIZ  /?/
 6524  033146  047111  040526  044514  INVLD:  .ASCIZ  /INVALID COMMAND/<15><12>
 6525  033154  020104  047503  046515
 6526  033162  047101  006504  000012
 6527  033170  020077  047111  040526  BADENT: .ASCIZ  /? INVALID ENTRY/<15><12>
 6528  033176  044514  020104  047105
 6529  033204  051124  006531  000012
 6530  033212  005015  051120  043517  INTDON: .ASCIZ  <15><12>/PROGRAM INITIALIZATION COMPLETE, ENTER COMMANDS/<15><12>
 6531  033220  040522  020115  047111
 6532  033226  052111  040511  044514
 6533  033234  040532  044524  047117
 6534  033242  041440  046517  046120
 6535  033250  052105  026105  042440
 6536  033256  052116  051105  041440
 6537  033264  046517  040515  042116
 6538  033272  006523  000012
 6539                          .EVEN
 6540
 6541  033276  033571          PARLST: .WORD   PAR11          ;PARAMETER ENTRY LIST
 6542  033300  001216                  .WORD   RPADR
 6543  033302  033600                  .WORD   PAR12
 6544  033304  001220                  .WORD   RPVEC
 6545  033306  033501                  .WORD   PAR1
 6546  033310  001306                  .WORD   BUFRSZ         ;LOCATION
 6547  033312  033531                  .WORD   PAR4
 6548  033314  001324                  .WORD   LC
 6549  033316  033535                  .WORD   PAR5
 6550  033320  001326                  .WORD   FC
 6551  033322  033541                  .WORD   PAR6
 6552  033324  001330                  .WORD   LT
 6553  033326  033545                  .WORD   PAR7
 6554  033330  001332                  .WORD   FT
 6555  033332  033551                  .WORD   PAR8
 6556  033334  001334                  .WORD   LS
 6557  033336  033555                  .WORD   PAR9
 6558  033340  001336                  .WORD   FS
 6559  033342  033511                  .WORD   PAR2           ;PARAMETER NAME
 6560  033344  001312                  .WORD   INTRVL
 6561  033346  033626                  .WORD   PAR19
```

```
                                        .WORD   PASCNT
                                        .WORD   PAR10
                                        .WORD   SEKMOD
                                        .WORD   PAR14
                                        .WORD   RATIO
                                        .WORD   PAR3
                                        .WORD   CMPLMT
                                        .WORD   PAR15
                                        .WORD   AUTOCK
                                        .WORD   0               ;TABLE TERMINATOR

                        ASKPAR: .ASCIZ  /ENTER PARAMETERS:/<15><12><12>

                        NGPAR:  .ASCIZ  <15><12>/ERROR IN PARAMETER:  /

                        REPAR:  .ASCIZ  /  RE-ENTER PARAMETERS/<15><12>

                        PAR1:   .ASCIZ  /BUFRSZ /
                        PAR2:   .ASCIZ  /INTRVL /
                        PAR3:   .ASCIZ  /CMPLMT /
                        PAR4:   .ASCIZ  /LC /
                        PAR5:   .ASCIZ  /FC /
                        PAR6:   .ASCIZ  /LT /
                        PAR7:   .ASCIZ  /FT /
                        PAR8:   .ASCIZ  /LS /
                        PAR9:   .ASCIZ  /FS /
                        PAR10:  .ASCIZ  /SEKMOD /
                        PAR11:  .ASCIZ  /RPADR /
                        PAR12:  .ASCIZ  /RPVEC /
                        PAR14:  .ASCIZ  /RATIO /
                        PAR15:  .ASCIZ  /AUTOCK /
                        PAR19:  .ASCIZ  /PASCNT /

                                .EVEN

                        CYLDER: .WORD   0,0,0           ;HEADER BUFFER FOR 'READHD' ROUTINE

                        ENDPGM  =       .               ;BUFFERS START HERE
```

```
                                    
033644  005015  046412  044501  TITLE:  .ASCII  <15><12><12>/MAINDEC-11-DZRP1-B/<15><12>
033652  042116  041505  030455
033660  026461  055104  050122
033666  026461  006502     012
033673     122  030520  042461          .ASCIZ  /RP11E MULTI-DRIVE EXERCISER PROGRAM/<15><12><12>
033700  046440  046125  044524
033706  042055  044522  042526
033714  042440  042530  041522
033722  051511  051105  050040
033730  047522  051107  046501
033736  005015  000012
                                        .EVEN
                                    .SBTTL  ROUTINE TO SIZE MEMORY
                                    ;;*****************************************************************
                                    ;*CALL:
                                    ;*      JSR     PC,$SIZE
                                    ;*      RETURN
                                    ;*SLSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
                                    
033742  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
033744  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
033746  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
033752  013746  000006                  MOV     @#ERRVEC+2,-(SP)
033756  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
                                    ;;SET THE ERRVEC PS TO THE PRESENT PS
033760  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
033762  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
033766  012737  034006  000004          MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
033774  012701  020000                  MOV     #20000,R1       ;;FIRST ADDRESS
034000  005711                  1$:     TST     (R1)            ;;TEST THIS ADDRESS
034002  005721                          TST     (R1)+           ;;STEP TO NEXT ADDRESS
034004  000775                          BR      1$              ;;TRY ANOTHER
034006  162701  000002          2$:     SUB     #2,R1           ;;DROP BACK
034012  010006                          MOV     R0,SP           ;;RESTORE THE STACK
034014  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
034020  012637  000004                  MOV     (SP)+,@#ERRVEC
034024  010137  034036                  MOV     R1,$LSTAD       ;;LAST ADDRESS
034030  012601                          MOV     (SP)+,R1        ;;RESTORE R1
034032  012600                          MOV     (SP)+,R0        ;;RESTORE R0
034034  000207                          RTS     PC
034036  000000                  $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
                                    
000001                                  .END
```

```
ABNRML  011306          2110    2963#
ACTDRV  021130          4690#   4834*   4843*   4974*   4984*   5105
ACTSTR  021131          4696#   5107*   5127*
AIE     = 020000        1212#
ASGN0   033131          2016    6521#
ASGN0   010430          2801#   2900    2906
ASGN1   010444          2805#
ASGN2   010502          1924    2804    2813#   2816
ASGN3   010524          2810    2813    2818#   2839
ASGN4   010626          2817    2819    2836#
ASGN5   010636          2837    2840#
ASGN6   010644          2825    2842#
ASGN7   010654          2826    2844#
ASGN8   010662          2841    2843    2845#
ASKPAR  033374          1765    6573#
ASNERR  011120          2811    2867    2885    2910#
ASNLST  025460          1719    1966    2023*   2549    2557    2808    2818    2854    2856*   2878    2888    2947*   3022*
                        5666#
ASNMSG  011140          2806*   2842*   2844*   2866*   2884*   2916#
ATABIT  021304          1891    2023    2549    2557    2808    2818    2854    2856    2878    2947    3022    4741#   4769
                        4816    4818    4896    4918    4919    4945    4991    4999    5068    5069    5076    5083    5096
                        5149    5185    5186
AUTOCK  001322          1350#   2299    2402    5023    6570
AVAIL   025550          1969    2017    2021*   2024    2044    2046    2073    2074    2113    5680#
BADENT  033170          6527#
BADOP   013606          3435    3453#
BADPAR  003256          1824    1831    1835    1839    1843    1847    1851    1855    1859#
BDADR   014106          3494    3527#
BEGCOD  001342          1376#   2028
BEGIN   002032          1669    1671#
BEGPAT  001340          1375#   2029
BEGSIZ  001344          1378#   1825    1827*   2031    2032
BIT0    = 000001        1128#   5226
BIT00   = 000001        1118#   1128
BIT01   = 000002        1117#   1127
BIT02   = 000004        1116#   1126    2126    2239    2409    3270    3547    3709    5025
BIT03   = 000010        1115#   1125
BIT04   = 000020        1114#   1124
BIT05   = 000040        1113#   1123
BIT06   = 000100        1112#   1122    3430    3472    5018
BIT07   = 000200        1111#   1121    3422    4878    4997    5093
BIT08   = 000400        1110#   1120    3428    5153
BIT09   = 001000        1109#   1119    3438    5175
BIT1    = 000002        1127#
BIT10   = 002000        1108#   3865
BIT11   = 004000        1107#
BIT12   = 010000        1106#   3436    4966    5016    5085    5159    5177
BIT13   = 020000        1105#   3434    3872    5246
BIT14   = 040000        1104#   3432    4964    5014    5088    5157    5180
BIT15   = 100000        1103#   4964    4966    5014    5016    5018    5085    5088    5091    5153    5175    5177    5180
                        5246
BIT2    = 000004        1126#
BIT3    = 000010        1125#
BIT4    = 000020        1124#
BIT5    = 000040        1123#   5091
BIT6    = 000100        1122#
```

DZRP1B.CMB CROSS REFERENCE TABLE -- USER SYMBOLS

```
BIT7  = 000200  1121# 3788  5091
BIT8  = 000400  1120# 3440
BIT9  = 001000  1119#
BLKADR  025736  1721  2828  2829  2858  2881  5710#
BLK0  024274  2548  5334#  5711
BLK1  024406  5374#  5712
BLK2  024520  5414#  5713
BLK3  024632  5454#  5714
BLK4  024744  5494#  5715
BLK5  025056  5534#  5716
BLK6  025170  5574#  5717
BLK7  025302  5614#  5718
BPTVEC=  000014  1135#
BUFRSZ  001306  1333#  1753  1755*  1756  1758*  1819  1821  1825  1827  2247  2352  2460  2461
                6546
BUFTBL  025614  1740*  1741*  1742*  1743*  1745*  1750*  1752*  1756  1758  1819  2149  2151  2169*
                2185  2186  2198*  2203*  2204  2212  2213  2226*  5688#
CHNGPM  001262  1316#  1668*  1670*  1759  1818*
CI  021646  4840  4849#  4982  5125
CIR  022436  4891  4913  4934  4960#
CI1  021760  4855  4875#  4882
CI2  022024  4858  4887#  4904
CI3  022142  4861  4909#  4926
CI4  022264  4864  4931#  4955
CKADR  002710  1760  1799#
CKBUS  012466  3234  3253#
CKCLK  006762  1905  2519#  3431  3473
CKCLK1  007044  2520  2530#
CKCLK2  007112  2531  2539#
CKCLK3  007116  2529  2538  2540#
CKERR  012434  3233  3242#
CKHNF  014460  3506  3609#
CKPAR  003076  1826  1828#
CKSWR =  104407  3640  3680  4626#
CK.CHR  011774  3095#  3121  3138
CK.NUM  012020  1779  3117#
CK.OCT  011744  3076#  3119  3133
CLEAR =  000001  1161#  5009  5160  5161
CLKFLG  001236  1305#  2519*  2523*  2533*  2684
CLOCK  007660  2525  2535  2693#
CLROPB  011150  2830  2922#
CMBUF  013164  3281*  3352#  6342
CMCNT  013154  3274*  3275*  3287  3289  3292*  3327  3348#
CMCYL  013156  3276*  3337*  3349#  6342
CMDAT  012712  3293#
CMPAR  012534  3237  3268#  3584  3674
CMPARD  012556  3271  3273#
CMPLMT  001314  1339#  3279  6568
CMPRES  006650  1994  2076  2095  2117  2494#  2497
CMPRT  013166  3305  3314  3324  3356#
CMPRT1  013274  3367  3373  3376#
CMPRX  013140  3328  3339#
CMSEC  013160  3278*  3329*  3330  3332*  3350#  6342
CMSTR  012620  3279#  3331  3335  3338
CMTRK  013162  3277*  3333*  3334  3336*  3351#  6342
CNTRLC  001242  1307#  1764*  1908*  4143  4226
```

```
CNTRLR  014116          3497    3533#
COMTBL  025756          2318    2413    5720#
CR      =  000015       1043#   3944    3954
CRLF    =  000200       1044#   3915    3954
CSME    =  000040       1191#   3507    3661
CYLDER  033636          3615    5654    6372    6615#
CYL.ER  001266          1318#   3848#   6364
DATAPK  011074          2788    2899#
DATAO   026030          5751    5754#
DATAI   026070          5736    5737    5738    5739    5740    5741    5742    5743    5744    5745    5746    5747    5748
                        5749    5750    5752    5771#
DATER   014622          3509    3637#
DDISP   =  177570       1050#   1271    1691
DEASGN  010664          2778    2849#
DEASSG  033067          1990    6514#
DF1     032264          1404    6375#
DF10    032274          1453    6384#
DF12    032304          1467    6391#
DF13    032314          1474    6398#
DF14    032320          1481    6402#
DF15    032324          1488    1544    1650    6406#
DF16    032330          1495    6410#
DF17    032334          1502    1509    1516    1523    1615    1643    6414#
DF2     032270          1411    1418    1460    6380#
DF23    032344          1530    6421#
DF24    032354          1537    6428#
DF26    032360          1551    6432#
DF27    032364          1558    1579    1586    1600    1657    6436#
DF34    032374          1593    1608    6443#
DH1     030352          1402    6190#
DH10    030412          1451    1465    6197#
DH10A   030476          6206#   6387
DH12    030517          6209#   6394
DH14    030571          1479    6217#
DH15    030617          1486    1648    6221#
DH16    030654          1493    6226#
DH17    030775          1500    1507    1514    1521    1613    1641    6240#
DH17A   031052          6248#   6417
DH2     030360          1409    6191#
DH23    031117          1528    6255#
DH23A   031222          6267#   6424
DH24    031237          1535    6270#
DH25    031304          1542    6277#
DH26    031341          1549    6282#
DH27    031416          1556    1577    1584    1598    1655    6290#
DH27A   031503          6299#   6439
DH3     030375          1416    6194#
DH30    031570          1563    6308#
DH31    031603          1570    6310#
DH34    031616          1591    1606    6312#
DH34A   031673          6320#   6446
DH41    031745          1627    6328#
DH42    031765          1634    6331#
DISPLA  001142          1271#   1691#   1699#   3864#
DISPRE  000174          1223#   1699
DONE    013662          3424    3478#
```

```
DORTI   006640          1908    2489#
DRIVE   001264          1317#   3843*   6339    6342    6346    6348    6351    6355    6358    6360    6362    6364    6370
DRNUM   033103          2953    2994    3000    6517#
DROP    011226          2944#   2967    3449    3460    3466    3474
DROPNG  033001          2952    6503#
DRVACT  021110          4649#   4867    4896*   4918*   4945*   4991*   5046    5052    5068*   5149*   5185*
DRVINT  021444          4778    4800#   4960    5010    5152
DRVQUE  024226          5029    5080    5249    5264#
DRVSTA  021112          1881    2824    4656#   4801*   4811*   4813*   4961    5011    5086*   5089*   5154    5178*   5181*
DRVTYP  021122          1811    1891    4670#   4816*   4818*
DSKERR= 000001          1186#   3571    3588
DSWR    = 177570        1049#   1270    1690
DT1     032010          1403    6338#
DT10    032026          1452    1466    6342#
DT11    032022          1459    6341#
DT13    032040          1473    6344#
DT14    032046          1480    6345#
DT15    032050          1487    1649    6346#
DT16    032060          1494    6348#
DT17    032076          1501    1508    1515    1522    1614    1642    6351#
DT17A   032112          6353#
DT2     032012          1410    6339#
DT23    032124          1529    6355#
DT23A   032140          6357#
DT24    032144          1536    6358#
DT25    032156          1543    6360#
DT26    032166          1550    6362#
DT27    032202          1557    1578    1585    1599    1656    6364#
DT27A   032220          6367#
DT3     032016          1417    6340#
DT34    032236          1592    1607    6370#
DT34A   032252          6372#
DUNIT   025504          1960    2858*   2949*   3024*   5672#
EMTVEC= 000030          1138#   1680*   1681*   1702*
EM1     027030          1401    6032#
EM10    027161          1450    6051#
EM12    027226          1464    6059#
EM15    027252          1485    6064#
EM16    027325          1492    6073#
EM17    027401          1499    6082#
EM2     027072          1408    6039#
EM20    027445          1506    6090#
EM21    027501          1513    6096#
EM22    027517          1520    6100#
EM23    027534          1527    6104#
EM24    027554          1534    6108#
EM25    027602          1541    6113#
EM26    027623          1548    6117#
EM27    027655          1555    6123#
EM3     027117          1415    6044#
EM32    027670          1576    6126#
EM33    027732          1583    6133#
EM34    030002          1590    6141#
EM35    030023          1597    6145#
EM36    030103          1604    6155#
EM37    030171          1612    6166#
```

```
EM43    030226          1640    6172#
EM44    030267          1647    6179#
EM45    030322          1654    6185#
ENDCMP  013362          3326    3392#
ENDCON  001274          1328#   2974    2977
ENDPAS  033035          2986    6508#
ENDPGM= 033644          1741    1743    2243    6617#
ENDSEK  001300          1330#   2980    2983
ENDTST  033051          2993    6511#
ENTPAR  002556          1764#   1806    1864
ENTPR   002564          1765#   1792
ENTPR1  002662          1764    1790#
ENTPR2  002674          1768    1784    1789    1793#
EOP     =  000002       1187#   2439    3478
ERCTR   013150          3284*   3301    3346#   3368*   3374*   3392    3691*   6345
ERPRC1  013476          3423    3428#
ERPROC  013462          3232    3422#
ERR     =  100000       1214#   3613    5074
ERRVEC= 000004          1131#   1688    1689*   1700*   1799*   1801*   1803*   2520*   2521*   2531*   2540*   6642    6643
                        6647*   6648*   6655*   6656*
FACTOR  004716          2122*   2123*   2125*   2128    2136#
FAIRNS  001254          1312#   2034    2047    2394    2470
FC      001326          1363#   1832    2027    2343    2347    2454    6550
FILBUF  005304          2058    2087    2238#   3660
FNTE    =  001000       1195#   3501    3571    3588
FNTERR  014136          3503    3545#
FRSTER  013143          3283*   3316    3342#   3356    3360*   3694*
FS      001336          1367#   1848    2025    2323    2327    2453    6558
FT      001332          1365#   1840    2026    2333    2337    2452    6554
FUV     =  040000       1200#   3486
GENDPB  025414          2822*   3749*   3750*   3752    3754    3761*   3762*   3763*   3764*   3765*   3766*   3768    3770
                        5654#
GENREG  025434          3613    5656    5658#
GETBUF  004720          2055    2083    2146#   3569    3656
GETCYL  005750          2342#
GETPAR  005512          2053    2292#
GETPAT  006304          2392    2418#   2423    2468
GETREM  011600          2308    2326    2336    2346    2354    2401    2419    3029#
GETRK   005720          2332#
GETSEC  005670          2313    2322#
GETSIZ  006000          2352#   2365
GNS     =  ******  U    1222    4618    4619    4620    4621    4622    4624    4626    4627    4628    4629    4630
GODRIV  005434          2060    2088    2270#   3782
GTPAR1  005566          2296    2303#
GTSWR   =  104406       1713    4624#
HDR     =  004000       1210#   3765
HE      =  040000       1213#   5074
HNF     =  010000       1179#   3242    3504
HOMSEK= 000015          1167#   3750    5232
HOUR    010130          1725*   2686    2715*   2716    2718    2720*   2721*   2723*   2735#
HT      =  000011       1041#   3913    3954
HZ      001240          1306#   1723    2696
IDE     =  000100       1207#   4943    5005
IDLE    004532          2045    2086    2100#
INCHRD  007544          2649#   3580    3669
INCMIS  007600          2667#   3626
```

```
INCSKI  007562          2658#  3603
INCSOF  007526          2640#  3583   3673
INCTOT  007616          2676#  3247   3262   3395   3448   3459   3465   3471   3516   3522   3528   3534   3540
                        3573   3587   3602   3632   3643
INTDUN  033212          1904   6530#
INTRFA  014126          3500   3539#
INTRVL  001312          1337#  1728*  2706*  2726   2728   2730*  6560
INVLD   033146          2794   6524#
IOTVEC= 000020          1136#
ISR     022502          4774   4974#
KIPARO= ****** U        6638
KSR     010146          1909   2747#  2761
KSR1    010230          2749   2760#
KSR2    010420          2756   2775   2779   2783   2789   2793   2795#
LC      001324          1359#  1815   1817*  1828   1832   2342   2447   6548
LF    = 000012          1042#  3948   3954
LIMIT   013152          3279*  3280*  3347#  3376   3378*
LINIIH  032404          3724   6451#
LODQUE  024060          4838   5225#
LPE   = 000200          1193#  3507   3661
LS      001334          1366#  1844   1848   2322   6556
LSTAD   001244          1308#  1739*  1742   1746
LT      001330          1364#  1836   1840   2332   2450   6552
MAIN    003650          1950#  2118   2459
MAIND   003676          1951   1953   1959#
MAIN1   004046          1926   1932   1967   1999#
MAIN2   004266          2009   2041#  2096
MAIN3   004446          2042   2081#
MASK    015572          3571*  3588*  3661*  3790   3802#
MATCH   013404          3293   3402#
MAXCYL  001256          1313#  1810*  1814*  1817   1828
MAXER   001310          1336#  2965
MINUTE  010134          1726*  2687   2707*  2708   2710*  2711*  2712   2714*  2737#
MODE  = 010000          1211#  3495   3765
MODERR= 000400          1194#
MONTR   003532          1914#
NEWASN  010424          2774   2800#
NEWUNT  025526          2003   2021   2022*  2828*  5676#
NGPAR   033421          1860   6577#
NOMTCH  015040          3295   3680#
NOTSAF  032615          1889   2844   6482#
NRML    011334          2111   2972#
NRMLX   011576          2979   2982   2984   3018   3025#
NRML1   011366          2973   2976   2980#
NRML2   011410          2455   2975   2978   2981   2985#
NXC   = 020000          1199#  3492
NXME  = 000004          1188#  3498
NXS   = 004000          1197#  3492
NXT   = 010000          1198#  3492
OFLIN   013574          3433   3447#
ORDERQ  025462          2061   2090   2100   5668#
PACK    001252          1311#  1730*  2800*  2832   2834   2899*  2905*
PARLST  033276          1766   6541#
PARXIT  006216          2302   2314   2394#
PAR1    033501          1823   6545   6588#
PAR10   033561          6563   6600#
```

```
PAR11    033571          6541    6602#
PAR12    033600          6543    6604#
PAR14    033607          1854    6565    6606#
PAR15    033616          6569    6608#
PAR19    033626          6561    6610#
PAR2     033511          6559    6590#
PAR3     033521          6567    6592#
PAR4     033531          1830    6547    6594#
PAR5     033535          1834    6549    6595#
PAR6     033541          1838    6551    6596#
PAR7     033545          1842    6553    6597#
PAR8     033551          1846    6555    6598#
PAR9     033555          1850    6557    6599#
PASCNT   001304          1332#   2991    6562
PATCOD   006206          2392#
PC     =%000007          1062#   1707*   1736*   1808*   1905*   1907*   1924*   1938*   1955*   1985*   1991*   1994*   2011*
                         2051*   2053*   2055*   2058*   2060*   2076*   2083*   2087*   2088*   2095*   2107*   2108*   2109*
                         2110*   2111*   2117*   2134*   2161*   2208*   2266*   2282*   2293*   2304*   2308*   2311*   2326*
                         2336*   2346*   2354*   2360*   2364*   2392*   2396*   2401*   2414*   2419*   2422*   2425*   2429*
                         2442*   2455*   2468*   2471*   2476*   2479*   2498*   2515*   2541*   2556*   2563*   2569*   2574*
                         2579*   2581*   2586*   2588*   2597*   2598*   2601*   2602*   2606*   2607*   2611*   2612*   2620*
                         2621*   2631*   2632*   2634*   2643*   2652*   2661*   2670*   2679*   2689*   2725*   2774*   2778*
                         2782*   2788*   2792*   2810*   2812*   2813*   2815*   2823*   2830*   2845*   2867*   2868*   2882*
                         2885*   2894*   2895*   2918*   2940*   2959*   2968*   3005*   3025*   3032*   3035*   3063*   3169*
                         3220*   3233*   3234*   3237*   3238*   3247*   3249*   3262*   3264*   3272*   3293*   3295*   3305*
                         3314*   3324*   3326*   3339*   3359*   3367*   3373*   3375*   3387*   3388*   3395*   3397*   3418*
                         3431*   3443*   3448*   3454*   3459*   3465*   3471*   3473*   3511*   3516*   3517*   3522*   3523*
                         3528*   3529*   3534*   3535*   3540*   3541*   3558*   3565*   3569*   3573*   3574*   3576*   3577*
                         3580*   3583*   3584*   3587*   3590*   3595*   3602*   3603*   3604*   3605*   3612*   3626*   3627*
                         3632*   3633*   3642*   3643*   3645*   3652*   3656*   3660*   3663*   3665*   3666*   3669*   3670*
                         3673*   3674*   3675*   3682*   3687*   3699*   3716*   3743*   3756*   3774*   3782*   3796*   3799*
                         3817*   3819*   3845*   3874*   3922*   3929*   3936*   3950*   3952*   4127*   4141*   4202*   4247*
                         4520*   4559*   4564*   4778*   4788*   4820*   4840*   4855*   4858*   4861*   4864*   4871*   4877*
                         4880*   4883*   4902*   4905*   4924*   4927*   4951*   4956*   4960*   4967*   4979*   4981*   4982*
                         5004*   5006*   5008*   5010*   5037*   5049*   5057*   5097*   5115*   5124*   5125*   5129*   5151*
                         5152*   5187*   5188*   5215*   5285*   6660*
PIRQ   = 177772          1048#
PIRQVE= 000240           1142#
POPQUE   024256          4880    4902    4924    4951    5281#   5284
PREVST   006616          2293    2442    2475#
PROCES   012374          2108    3230#
PROG   = 002000          1196#   3495
PRTBAD   015144          3642    3670    3703#
PR0    = 000000          1065#
PR1    = 000040          1066#
PR2    = 000100          1067#
PR3    = 000140          1068#
PR4    = 000200          1069#   1906    2748
PR5    = 000240          1070#
PR6    = 000300          1071#
PR7    = 000340          1072#
PS     = 177776          1045#   1046    1706*   4762    4763*   4783*   4786*   4844*
PSW    = 177776          1046#   1729*   1791*   1794*   1906*   1910*   4832    4833*
PWRVEC= 000024           1137#
QDONE    021222          4717#   4862    4931    4932    4933    4950    4954    5079
QREGS    021134          4708#   4853    4875    4876    4879    4881    5228
```

```
QSEEK   021156          4711# 4856  4887  4888  4890  4899  4901  4903  5234
QTRANS  021200          4714# 4859  4909  4911  4912  4922  4923  4925  5028  5248
QUES    033144          2910  6523#
QWAIT   021244          4720# 4899* 4922* 5059  5065* 5165  5183*
RAMWRT  006230          2311  2400#
RATIO   001320          1341# 1852  2309  6566
RDCHR = 104410          4302  4627#
RDLIN = 104411          1777  2760  4628#
RDSEK = 000005          1163# 2385  2465  5236  5723
RDY   = 000200          1208#
READ  = 000017          1168# 2372  3766  5242  5724
READOR  006666          2476  2502# 3558  3645  3845
READHD  015402          3612  3761#
RECALT  015350          3576  3604  3627  3665  3749#
RECALD  015354          2823  3750#
REDAPK  011106          2792  2905#
RELBUF  005056          2109  2184# 3565  3652
REPAR   033451          1863  6582#
RESREG= 104413          2207  2265  2568  2580  2795  3219  4563  4630# 4787  4819  4870  4983  5126
                        5214
RESVEC= 000010          1132#
RETRY   015472          3574  3577  3590  3663  3666  3781#
RPADR   001216          1297# 1800  4776  4850  4976  5138  5198  6338  6542
RPBA  = 000010          1150# 4937* 6348  6351  6358  6367
RPCA  = 000012          1151# 4893* 4915* 4938* 6353  6357  6362  6367  6372
RPCS  = 000004          1148# 4802* 4869* 4875* 4887* 4895* 4909* 4917* 4931* 4944* 4998* 5000  5005*
                        5009* 5044  5048* 5061* 5074  5139* 5147  5160* 5161* 5199* 6339  6346  6350
                        6351  6355  6358  6360  6362  6364  6370
RPDA  = 000014          1152# 4894* 4916* 4939* 6353  6362  6367  6370
RPDS  = 000000          1146# 4803  4805  4807  4809  4814  4999* 5041  5043  5062  5069* 5070  5072
                        5167  5169  5171  5173  6346  6348  6351  6355  6358  6360  6362  6364  6370
RPER  = 000002          1147# 3613  6346  6348  6351  6355  6358  6360  6362  6364  6370
RPERRS  021104          4642# 4764  5043* 5044* 6340
RPINIT  021314          1808  4761#
RPM1  = 000016          1153# 6353  6367
RPM2  = 000020          1154# 5203
RPM3  = 000022          1155#
RPPRIO  001222          1299# 4763  4775  4833
RPTMR   023360          2725  5105#
RPVEC   001220          1298# 4773  6544
RPWC  = 000006          1149# 4936* 6348  6351  6367
RPZERO  011656          3040  3042  3044  3046#
RPZRC   011742          3039* 3041* 3043* 3045* 3058  3065#
RP02    032503          1890  6466#
RP03    032510          1893  6467#
RP11    021566          2272  3751  3767  4832#
R0    =%000000          1053# 1779* 1805* 1930* 1933  1935  1938  1983* 1987  2024* 2025* 2026* 2027*
                        2028* 2029* 2030* 2031* 2032* 2033* 2034* 2046* 2047  2049  2056* 2059* 2064
                        2066* 2081* 2084* 2089* 2093  2101* 2103  2122  2123  2126  2128* 2129* 2130
                        2132* 2133* 2152  2163  2165* 2166  2167* 2188  2190  2196  2197  2201  2202
                        2209  2210  2239  2241  2243  2247  2251  2252  2254  2270  2271  2272* 2275*
                        2276* 2277* 2278  2280* 2281* 2294* 2297  2301* 2303* 2312  2318* 2328* 2338*
                        2348* 2366  2369* 2370  2372  2375  2379  2381  2383* 2385* 2387* 2388* 2393*
                        2394* 2409  2413* 2430  2432  2434  2439  2441* 2443* 2444* 2445* 2446* 2447
                        2450  2452* 2453* 2454* 2460* 2461* 2462* 2463  2465* 2467* 2469* 2470* 2477*
                        2478* 2503  2509  2514  2548* 2552* 2559* 2561  2564# 2576  2577  2596  2626
```

```
                    2627    2628    2629    2630    2640    2642*   2649    2651*   2658    2660*   2667    2669*   2676
                    2678*   2693    2732*   2829*   2831*   2834*   2881*   2926    2946    2949    2954    2965    2974
                    2977    2980    2983    2987    2991    2995    3001    3008    3017*   3021    3024    3046    3047*
                    3048    3050*   3051*   3053    3055*   3056*   3061*   3083    3084*   3099    3100    3102*   3103*
                    3119*   3121*   3133*   3138*   3146*   3148*   3149*   3159*   3160*   3161*   3162*   3163*   3164*
                    3165*   3166    3172    3177    3180*   3181    3185    3186    3202    3207    3210    3212*   3230
                    3242    3244    3253    3255    3257    3259    3268    3270    3273    3274    3275    3276    3277
                    3278    3422    3428    3432    3434    3436    3438    3440    3478    3480    3483    3486    3489
                    3492    3495    3498    3501    3504    3507    3545    3547    3549    3550    3554    3559*   3560*
                    3561*   3562    3566*   3567*   3570*   3599    3600    3609    3611    3622    3624    3628    3629*
                    3637    3646*   3647*   3648*   3649    3653*   3654*   3657*   3658    3707    3709    3711    3712
                    3713    3730    3749    3751*   3763    3764    3767*   3783    3788    3790    3843    3848    3850
                    3851    3852    3853    3854    3855    3856    3857    3858    3859    3905    3906*   3907    3910*
                    4046    4056*   4060    4076    4077    4090*   4386    4393*   4396*   4398*   4403*   4410*   4411
                    4414*   4421*   4422    4426*   4433    4437*   4459    4484*   4499    4502*   4505*   4509*   4512*
                    4515    4519*   4539*   4540    4558*   4562*   4595    4596*   4597    4598*   4599*   4600*   4601*
                    4836    4838*   4841    4845*   4891*   4913*   4934*   4969    4970*   5029*   5080*   5249*   5251
                    5253*   5271    5273*   6640    6644*   6654    6659*
R1      =%000001    1054#   1778*   1969*   1970    1972    1974*   1976*   1977    1979    1981*   1983    1993*   2074*
                    2075*   2094*   2100*   2101    2105*   2106    2112*   2116    2146    2151*   2152    2156*   2160*
                    2162    2163*   2166*   2173    2175*   2176*   2185*   2191    2193*   2196*   2197*   2201*   2202*
                    2209*   2210*   2217    2225    2227    2229*   2230*   2251*   2257*   2494*   2496*   2524*   2525*
                    2526*   2534*   2535*   2536*   2615    2616*   2623*   2625*   3076    3078    3080    3095    3097
                    3101*   3125*   3177*   3189    3193    3273*   3281    3299    3311    3319    3363    3365*   3372*
                    3382    3385    3402    3404*   3410    3417*   3684    3685    3690*   3711*   3720*   3722*   3728
                    3730    3733    3841    3849*   3850*   3851*   3852*   3853*   3854*   3855*   3856*   3857*   3858*
                    3859*   3861*   4047    4060*   4061    4065    4089*   4387    4392*   4397*   4415*   4418    4422*
                    4431*   4432    4436*   4460    4483*   4500    4503*   4506*   4510*   4511*   4513*   4514*   4516
                    4518*   4541*   4547*   4553*   4764*   4766*   4767    4770*   4771    4777*   4779*   4780*   4782*
                    4784*   4801*   4802    4811*   4813*   4816    4818    4889*   4890*   4896    4897*   4898*   4899*
                    4900*   4901*   4910*   4911*   4918    4919    4920*   4921*   4922*   4923*   4932*   4945    4946*
                    4947*   4948*   4950*   4961    4989*   4990*   4991    4992*   4993*   4994*   4998    4999    5011
                    5027    5028*   5036*   5050*   5054*   5058*   5059    5060*   5061    5064*   5065*   5066*   5067*
                    5068    5069    5076    5078    5079*   5082*   5083    5086*   5089*   5096    5109*   5117*   5119
                    5139    5140*   5141*   5142*   5145    5149    5154    5164*   5165    5166*   5178*   5181*   5182*
                    5183*   5184*   5185    5186    5225    5228*   5234*   5248*   5252*   5265    5270*   5281*   5283*
                    6641    6649*   6650    6651    6653*   6657    6658*
R2      =%000002    1055#   1785    1788    2000*   2017    2019*   2021*   2024    2035*   2043*   2044    2046    2068
                    2073    2075    2147    2149*   2154*   2159*   2171*   2177*   2186*   2194*   2199*   2211    2212*
                    2220*   2224*   2231*   2252*   2258*   2561*   2562*   2577*   2578*   2600    2604*   2605    2609*
                    2610    2614*   2617*   2618    3080*   3081*   3132    3135*   3181*   3199*   3289*   3291*   3292
                    3306*   3691    3693*   3727*   3735*   3842    3860*   4048    4059*   4063*   4066    4073*   4074*
                    4075    4080*   4088*   4388    4399*   4407*   4412    4435*   4461    4482*   4501    4504*   4507*
                    4517*   4538*   4541    4542*   4548*   4549*   4554*   4555*   4765*   4767    4769*   4771    4835
                    4836*   4837*   4842*   4876*   4878*   4879*   4888*   4893    4894    4895    4912*   4915    4916
                    4933*   4936    4937    4938    4939    4940    4941    4942    4949    4964*   4966*   4995*   4997*
                    5014*   5016*   5018*   5025    5031*   5032*   5033*   5034*   5035*   5059*   5085*   5088*   5091*
                    5093*   5150*   5153*   5157*   5159*   5165*   5175*   5177*   5180*   5199    5200*   5210*   5226
                    5230    5232    5236    5238    5240    5242    5244    5246*   5270
R3      =%000003    1056#   1874*   1900*   1916*   1923*   1959*   1964*   1999*   2007*   2148    2158*   2173*   2174*
                    2175    2176    2188*   2189*   2190*   2191    2256*   2260*   2262*   2547*   2553*   2566*   2802*
                    2807*   2836*   2852*   2860*   2869*   2887*   2891*   2923    2928*   2930*   2933*   2935*   2939*
                    3006    3010*   3013*   3016*   3117    3118*   3126*   3128*   3130*   3132*   3135    3136*   3142
                    3143    3144    3145    3146    3147*   3195*   3195*   3298*   3308*   3409*   3412*   3680    3683*
                    3688*   3695*   3988    3997*   4003*   4004*   4007*   4012*   4013*   4014    4023*   4049    4057*
                    4058*   4072*   4075*   4084*   4085*   4087*   4297    4299*   4300    4303*   4304    4311*   4312
```

```
                4314    4322    4326    4328*   4334    4336    4338*   4341*   4389    4411*   4412*   4414    4434*
                4462    4481*   4546*   4551*   4557*   4558    4773*   4774*   4775*   5051*   5052    5055*   5110*
                5111    5113*   5118    5202*   5204    5206    5207    5209    5211    5212
R4      =%000004  1057#  1766*   1767    1773    1786    1873*   1877    1881    1891    1902*   1917*   1922*   2001*
                2006*   2013    2023    2214*   2215*   2216*   2217    2253*   2254*   2255    2263    2546*   2549
                2551*   2557    2565*   2575*   2576*   2592    2801*   2805*   2808    2818    2822    2824    2827*
                2828*   2829    2835*   2838*   2849*   2853*   2854    2856    2857*   2858*   2859*   2862*   2874*
                2877*   2878    2880*   2881    2890*   2912    2924    2926*   2927*   2929*   2932*   2934*   2938*
                2944    2945*   2946*   2947    2948*   2949*   2958*   3007    3008*   3009*   3012*   3015*   3020*
                3021*   3022    3023*   3024*   3186*   3187*   3297    3403*   3405*   3406    3414*   3989    3991*
                3992*   3993*   3994    3995*   4009    4011*   4019*   4022*   4463    4480*   4544*   4547    4553
                4555    4776*   4802*   4803    4805    4807    4809    4814    4850*   4869*   4875*   4887*   4893*
                4894*   4895*   4909*   4915*   4916*   4917*   4931*   4936*   4937*   4938*   4939*   4944*   4976*
                4998*   4999*   5000    5005*   5009*   5041    5043    5044    5048*   5061*   5062    5069*   5070
                5072    5074    5138*   5139*   5147    5160*   5161*   5167    5169    5171    5173    5198*   5199*
                5208
R5      =%000005  1058#  1719*   1720*   1721    1773*   1774    1785*   1788*   1960*   1961    1963*   1972    1979
                1992*   2002*   2003    2005*   2021    2022*   2061*   2062    2064*   2070*   2071    2073*   2090*
                2091    2093*   2113*   2114    2116*   2213*   2214    2216    2219*   2225*   2227*   2228*   2229
                2230    2255*   2257    2263*   2292    2307*   2309    2315*   2316*   2317*   2318    2322*   2323*
                2325*   2327*   2328    2332*   2333*   2335*   2337*   2338    2342*   2343*   2345*   2347*   2348
                2352*   2353*   2355    2357    2368*   2369    2393    2395*   2400*   2405*   2407    2411*   2413
                2418*   2420    2424*   2469    2762*   2763*   2765    2767    2769    2771*   2772    2776    2780
                2784    2790    2803    2805    2850    2853    2864    2875    2877    2925    2937*   3031    3033*
                3179*   3183*   3204*   3215    3297*   3299    3320    3364    3366*   3371*   3384    3406*   3410
                3990    3996*   3998*   4000*   4001*   4002*   4003    4021*   4050    4052*   4054*   4061*   4065*
                4080    4086*   4464    4479*   4545*   4549    4556    5203*   5204
R6      =%000006  1059#  1061    1674*   1675*   1676
R7      =%000007  1060#  1062
SAVEFG  021132  1809*   4704#   5002    5094
SAVER1  013144  3285*   3319*   3344#   3361    3365    3369*
SAVER5  013146  3286*   3320*   3345#   3366    3370*
SAVREG= 104412  2184    2230    2545    2573    2747    3156    4537    4629#   4761    4800    4849    4975    5108
                5197
SC      023034  4981    5041#
SCMND   010772  2782    2874#
SCO     023076  5007    5015    5017    5019    5042    5050#   5124
SC1     023104  5052#   5056
SC2     023112  5054#   5063    5084    5095    5098
SC3     023122  5053    5058#
SDETAL  007306  2563    2579    2592#
SECOND  010140  1727*   2688    2698*   2699    2701*   2702*   2703    2705*   2739#
SEC.ER  001272  1320#   3847*   6364
SEEK    =  000011  1165#  4917    5230
SEEKWT  021126  4684#   4919*   5076    5083*   5096*   5186*
SEKMOD  001316  1340#   2972    6564
SETVEC  003302  1853    1869#
SHDTYP  007270  2556    2574    2585#
SILO    =  000026  1157#  5212
SIXTEE  010144  1723*   1724*   2694*   2696*   2697*   2743#
SIZMEM  002370  1734#   1737*   1738*
SKIER   014424  3491    3599#
SP      =%000006  1061#  1678*   1688*   1696*   1700    1774*   1778    1790*   1795*   1823*   1830*   1834*   1838*
                1842*   1846*   1850*   1854*   1859    1877*   1967*   2013*   2054*   2056    2082*   2084    2106*
                2112    2146*   2147*   2148*   2158    2159    2160    2162*   2211*   2222    2224    2270*   2275
                2292*   2357*   2358*   2359*   2361    2363*   2374*   2375*   2376*   2377    2386*   2395    2458
```

```
                    2475* 2477  2478  2489* 2502* 2503* 2504* 2505* 2507* 2508* 2509* 2510* 2512*
                    2514* 2530* 2539* 2592* 2596* 2600* 2605* 2610* 2615* 2618* 2625  2626* 2627*
                    2628* 2629* 2630* 2693* 2724* 2732  2748* 2749* 2750* 2751* 2752  2756* 2762
                    2816* 2840* 2912* 2923* 2924* 2925* 2937  2938  2939  2944* 2954* 2958  2987*
                    2995* 3001* 3006* 3007* 3015  3016  3029* 3030* 3031* 3033  3034  3046* 3047
                    3057  3061  3062* 3117* 3147  3189* 3193* 3248* 3253* 3254* 3255* 3256  3257
                    3263* 3363* 3364* 3371  3372  3396* 3402* 3404  3416* 3417  3549* 3550* 3551
                    3557* 3559  3560  3568* 3570  3609* 3610* 3611* 3644* 3646  3647  3655* 3657
                    3680* 3695  3712* 3713* 3714* 3715* 3717  3719* 3720  3723* 3728* 3733* 3761
                    3762  3772* 3773  3795* 3798  3815  3816* 3818* 3841* 3842* 3844* 3846  3847
                    3860  3861  3869  3905* 3906  3907* 3909  3910  3911* 3913  3915  3917  3923
                    3925* 3927* 3935* 3939  3943  3944  3948  3980* 3981  3982  3983* 3988* 3989*
                    3990* 3996  4021  4022  4023  4024* 4025* 4046* 4047* 4048* 4049* 4050* 4051*
                    4052  4055* 4068  4070* 4072  4082  4084  4086  4087  4088  4089  4090  4092*
                    4093* 4136* 4137* 4138  4142  4144  4153  4155  4158  4161* 4162* 4163  4168
                    4170  4172* 4173  4189* 4190* 4191  4201  4208* 4211* 4212* 4216* 4217* 4219
                    4222* 4229  4232* 4236  4238  4240  4241* 4248  4250  4252* 4253  4255* 4256*
                    4257* 4258* 4259* 4274* 4275* 4276* 4277* 4278* 4284* 4297* 4298* 4303  4306
                    4310* 4317  4321* 4340  4341  4342* 4343* 4344* 4385* 4386* 4387* 4388* 4389*
                    4390* 4391* 4392  4393  4395* 4399  4402* 4405* 4416* 4420* 4423  4424  4427*
                    4428* 4429  4432* 4433* 4434  4435  4436  4437  4438* 4459* 4460* 4461* 4462*
                    4463* 4464* 4465* 4466* 4467* 4468* 4475* 4476* 4477* 4478* 4479  4480  4481
                    4482  4483  4484  4499* 4500* 4501* 4517  4518  4519  4538  4540* 4595* 4596
                    4606* 4607* 4762* 4786  4832* 4835* 4842  4844  4940* 4941* 4942* 4943* 4944
                    5027* 5036  5078* 5082  5113  5128* 5208* 5209* 5210  5225* 5252  5264* 5267*
                    5272  6640* 6641* 6642* 6643* 6644  6647  6654* 6655  6656  6658  6659
SPACE1  032501      1776  1887  2613  2622  6465#
SPACE2  032500      2599  2603  2608  3197  3206  3217  3732  6464#
SPACE3  032477      2595  6463#
SPACE4  032476      1880  1896  6462#
STACK = 001100      1036# 1309  1678  1790  1795
STACK1  001246      1309#
START   002016      1241  1668# 1942
START1  002024      1242  1670#
STATHD  032646      2587  6487#
STATIN  001250      1310# 1717* 1950  1954* 2731*
STATIS  004626      2107  2122#
STATPR  007126      1955  2545# 2894
STKLMT= 177774      1047#
STNDAT  025764      2255  2263  3406  3414  5735#
STO     023472      5115  5138#
STO1    023636      5144  5164#
STO2    023716      5168  5177#
STO3    023734      5172  5180#
STO4    023746      5176  5179  5182#
STO5    023776      5146  5148  5163  5170  5174  5188#
SUCA    = 000024    1156#
SUFU    = 001000    1176# 3242  4803  5070  5167
SUOL    = 040000    1181# 4805  5072  5171
SUROY   = 100000    1182# 4809  5173
SURPO3= 020000      1180# 4814
SUSI    = 004000    1178# 3242  3489  4807  5169
SUSU    = 002000    1177# 5062
SUWP    = 000400    1175#
SVRP11  024000      4877  4967  5004  5008  5097  5151  5187  5197#
SWR     001140      1270# 1676  1690* 1692  1698* 1711  1871  1952  2295  2305  2437  2456  2754
```

```
                    2786    2963    2989    3157    3235    3303    3380    3696    3703    3705    3725    3740    3865
                    3872    3877    4146    4185    4240#
SWREG   000176      1224#   1698    1711    2754    4146    4185    4208
SW0     = 000001    1100#   2305    2786
SW00    = 000001    1090#   1100
SW01    = 000002    1089#   1099    3235
SW02    = 000004    1088#   1098    3380
SW03    = 000010    1087#   1097    3725
SW04    = 000020    1086#   1096    2456    2963    2989
SW05    = 000040    1085#   1095    2295    2437
SW06    = 000100    1084#   1094    1952
SW07    = 000200    1083#   1093    1871
SW08    = 000400    1082#   1092
SW09    = 001000    1081#   1091
SW1     = 000002    1099#
SW10    = 002000    1080#
SW11    = 004000    1079#
SW12    = 010000    1078#
SW13    = 020000    1077#   3157    3705
SW14    = 040000    1076#
SW15    = 100000    1075#   3696    3740
SW2     = 000004    1098#
SW3     = 000010    1097#   3303    3703
SW4     = 000020    1096#
SW5     = 000040    1095#
SW6     = 000100    1094#
SW7     = 000200    1093#
SW8     = 000400    1092#
SW9     = 001000    1091#
SYSTAT  032625      1876    6484#
TBITVE= 000014      1133#
TD      022552      4979    4989#
TIMEE   = 000020    1190#   3498
TIMER   021264      4728#   4765    4898#   4921#   4947#   4993#   5066#   5111    5113#   5141#
TIMIO   013636      3441    3470#
TIMOUT  001260      1315#   4898    4921    4947
TIMPOS  013624      3439    3464#
TITLE   033644      1704    6619#
TKVEC   = 000060    1140#   1909#   2758#   2761#   4123#   4124#
TPVEC   = 000064    1141#
TRAPVE= 000034      1139#   1682#   1683#   1703#
TRK.ER  001270      1319#   3846#   6364
TRNSWT  021124      2820    4676#   4851    4865    4949#   4952    4977    4990    4995    4996#   5122    5143    5145
                    5150    5162#
TRTVEC= 000014      1134#
TYPDS   = 104405    2988    3194    4622#
TYPE    = 104401    1704    1765    1770    1771    1776    1860    1861    1863    1875    1876    1880    1894    1896
                    1897    1899    1904    1984    1986    1990    2010    2012    2016    2585    2587    2595    2599
                    2603    2608    2613    2622    2633    2686    2687    2688    2764    2794    2910    2911    2915
                    2917    2950    2951    2952    2953    2957    2985    2986    2993    2994    2998    3000    3004
                    3019    3059    3168    3170    3174    3175    3184    3197    3201    3206    3208    3214    3217
                    3724    3732    3737    3739    3867    3875    3918    4015    4091    4140    4152    4206    4207
                    4210    4221    4231    4242    4261    4309    4315    4320    4324    4329    4330    4332    4335
                    4339    4618#
TYPERR  012130      3156#   3359    3387    3682    3687    3874
TYPEST  007240      1991    2573#   2882    3005
```

```
TYPOC = 104402  1775  3190  3729  3734  4209  4619#
TYPON = 104404  4621#
TYPOS = 104403  1878  1988  2014  2593  2913  2955  2996  3002  4620#
UNSAF   013612  3437  3458#  3488
UNTASN  032567  2806  6478#
UNTMSG  032515  1986  2012  2911  6468#
UNTNOT  032545  2866  2884  6474#
UNTOFF  032524  1886  2842  6470#
UNTON   032535  1884  6472#
WAIT    025572  1976  2041  2070  2081  2094  5684#
WC      022745  5006  5023#
WCE   = 000010  1189#  3480  3637  3661
WCEER   014066  3482  3515#
WPE   = 000100  1192#  3507  3661
WPV   = 100000  1201#  3483
WPVER   014076  3485  3521#
WRITE = 000013  1166#  2370  5244  5721
WRTCHK= 000007  1164#  2241  2297  2312  3268  3545  3658  3707  5035  5240  5722
WRTPK   006332  2051  2429#
WRTPK1  006362  2431  2433  2437#
WRTPK2  006560  2436  2465#
WRTPK3  006570  2435  2464  2467#
WRTPK4  006606  2438  2466  2470#
WRTSEK= 000003  1162#  2383  2467  5238  5720
ZROIND  013142  3282*  3313*  3318*  3322  3341#
$AUTOB  001134  1267#  1715*  4199  4356
$BDADR  001122  1262#  3382*  3383*  3684*  6341  6344
$BDDAT  001126  1264#  3385*  3600*  3615*  3616*  3617*  3618*  3619*  3620*  3621*  3622  3685*  6341
        6344  6355
$BELL   001206  1292#  3867  3883  4152  4349
$BUF  = 000006  2056*  2084*  2123  2190  2196  2201  2209  2243  2251  3255  3273  3550  3554
        3570*  3657*  5299#  5300
$CHARC  016270  3920*  3930*  3937  3946*  3951#
$CKSWR  017244  4185#  4626
$CNTAG  001100  1250#  1673  1674  1686
$CN1  = 000012  1282#  1283#  1284#  1285#  1286#  1287#  1288#  1289#  1290#  1291#  1292#
$CN2  = 000024  1282#  1283#  1284#  1285#  1286#  1287#  1288#  1289#  1290#  1291#  1292#
$CN3  = 000012  1280#  1282
$CNTLC  020144  4140  4221  4349#
$CNTLG  020156  4206  4351#
$CNTLU  020151  4231  4324  4350#
$COMND= 000002  2028*  2126  2239  2241  2297  2301*  2303  2312  2318*  2366  2370  2372  2379
        2381  2383*  2385*  2413*  2441  2465*  2467*  3268  3270  3545  3547  3658  3707
        3709  3750*  3765*  3766*  5025  5032  5035*  5296#  5297
$CRLF   001213  1294#  1705  1770  1875  1899  1984  2010  2585  2633  2764  2917  2950  2951
        2957  2985  2998  3004  3019  3168  3174  3184  3201  3214  3737  3739  3875
        3883  3919  3954  4242  4329  4349
$CYL  = 000010  2027*  2130  2278  2348*  2446*  2447  2454*  3276  3561*  3600  3629*  3648*  3764*
        5033  5300#  5301
$DBLK   016736  4057  4091  4099#
$DB2D   020624  2601  2606  2611  3817  4537#
$DECVL  021004  4539  4585#
$DIV    020206  2360  3032  3716  4383#
$DOAGN  003644  1936  1942#
$DTBL   016726  4060  4095#
$ENDAD  003634  1232  1807  1933  1934  1938#
```

```
SERFLG  001103          1253#  3862*  3883
SERMAX  001115          1259#
SERROR  015624          1680   3839#
SERRPC  001116          1260#  3869*  3870*  3871   3883
SERRTB  001346          1397#  3165
SERTTL  001112          1257#  3868*  3883
$FAIR =  000064         2034*  2047   2059*  2066*  2089*  2394*  2470*  5319#  5320
$FILLC  001156          1278#  3923   3954
$FILLS  001155          1277#  3954
SGDADR  001120          1261#
SGDDAT  001124          1263#  3256*  3384*  3599*  3624*  6344   6348   6355   6370
SGET42  003602          1930#  1968
SGTSWR  017334          4207#  4624
SHARD =  000054         2628   2649   2651*  5315#  5316
SHD   =  000000         1014
SHINUM  020620          1684*  2315   3029   4503   4511   4516*  4521#
SICNT   001104          1254#
SINTAG  001135          1268#  4204*  4223   4243   4356
SITEMB  001114          1258#  3160   3358*  3386*  3681*  3686*  3871*  3883
SLF     001214          1295#  3883   3954   4339   4349
SLKCSB  001226          1301#  2527*
SLKCSR  001224          1300#  2522   2528*
SLKS    001232          1303#  2532   2537*
SLLVEC  001234          1304#  2534
SLONUM  020622          1685*  4502   4509   4515*  4522#
SLPADR  001106          1255#
SLPERR  001110          1256#
SLPVEC  001230          1302#  2524
SLSTAD  034036          1739   6657*  6661#
SMAIL =  ******  U      1702   1711   3877   3907
SMEX  =  000001         5295#  5296
SMISPO=  000060         2630   2667   2669*  5317#  5318
SMNEM   020174          4210   4354#
SMODE =  000003         5297#  5298
SMSWR   020163          4207   4352#
SNULL   001154          1276#  3925   3954
SOCNT   016516          3987*  4016*  4029#
SOMODE  016520          3982*  3986*  3991   3994*  4005*  4031#
SOPERC=  000034         2276*  2277*  2430   2432   2562   2578   3009   5310#  5311
SPACK =  000022         2049   2434   2463   2834*  5306#  5307
SPASS   001100          1251#
SPASSC=  000062         2596   2831*  2987   2991   3017*  5318#  5319
SPATTC=  000026         2029*  2030*  2254   2393*  2469*  5308#  5309
SPOSIT=  000040         2132*  2133*  2280*  2281*  2980   2983   5311#  5312
SPREVA=  000030         2278   2294*  2443*  2477*  2478*  3599   3624   3628*  5033*  5034*  5309#  5310
SPREVO=  000024         2301   2303*  2409   2441*  5032*  5307#  5308
SQUES   001212          1293#  3883   3954   4261   4332   4349
SRAND   020522          2304   2364   2422   2429   4498#
SROCHR  017606          4274#  4627
SRODEC=  ******  U      4629
SROLIN  017676          4297#  4628
SROOCT=  ******  U      4629
SROSZ =  000012         4290#
SREAD =  000044         2128*  2129*  2974   2977   5312#  5313
SREG  =  000014         2933   5303#  5304
SREGAD  001160          1280#
```

F12


```
SREG0   001162          1282#  3849   6339   6346   6348   6350   6351   6353   6355   6357   6358   6360   6362
                        6364   6367   6370   6372
SREG1   001164          1283#
SREG10  001202          1290#
SREG11  001204          1291#
SREG2   001166          1284#
SREG3   001170          1285#
SREG4   001172          1286#
SREG5   001174          1287#
SREG6   001176          1288#
SREG7   001200          1289#
SRESRE  020464          4474#  4630
SRETRY  015570          3572*  3599*  3662*  3781*  3786*  3792*  3793   3801#
SRPBA = 000076          2122   3257   3549   3554   3711   3730   3854   5324#  5325
SRPCA = 000100          2446   3855   5325#  5326
SRPCS = 000072          3852   5322#  5323
SRPDA = 000102          2444   2503   2509   2514   3609   3611   3856   5326#  5327
SRPOS = 000066          3242   3489   3504   3850   5320#  5321
SRPER = 000070          2439   3244   3478   3480   3483   3486   3492   3495   3498   3501   3507   3637   3790
                        3651   5321#  5322
SRPM1 = 000104          3657   5327#  5328
SRPWC = 000074          3259   3275   3713   3853   5323#  5324
SRPZR0  011652          2612   3045#
SRPZR4  011632          2621   2632   3041#
SRPZR5  011642          2598   3043#
SRPZR8  011622          2602   2607   3039#
SR2A  = ******  U       4631
SSAVRE  020426          4458#  4629
SSB2D   015574          2597   2620   2631   3815#
SSEC  = 000012          2025*  2328*  2375   2444*  2445*  2453*  3278   3560*  3647*  3762*  5034   5301#  5302
SSETUP= 000146          1659#  1679   1680   1682   1684   1686   1708   3840   3880   3882   4144   4149   4150
                        4180   4356
SSILO = 000110          2552   2559   2564   2933   3859   5329#
SSIZE   033742          1736   6640#
SSKI  = 000056          2629   2658   2660*  5316#  5317
SSOFT = 000052          2627   2640   2642*  5314#  5315
SSTUP = 177777          1659#
SSUCA = 000106          2130   2294   2443   3561   3622   3628   3648   3764   3848   3858   5328#  5329
SSVPC = 000200          1230#  1235
SSWR  = 122000          1003#  1014   1018   1019   1020   1021   1292   1686   3832   3833   3834   3835   3836
                        3865   3872   3877   3881   3883
STATUS= 000016          2103   3230   3422   3428   3432   3434   3436   3438   3440   3754   3770   3783   3788
                        5304#  5305
STIME   007634          1985   2011   2586   2684#  3169
STKB    001146          1273#  2750   4104   4125   4136   4161   4189   4216
STKCNT  016746          4105#  4120*  4150   4167*  4281   4283*
STKINT  016756          1707   1907   4120#  4141   4202
STKQEN= 016755          4109#  4175   4286
STKQIN  016750          4106#  4121*  4122   4173*  4174*  4175   4177*
STKQOU  016752          4107#  4122*  4284   4285*  4286   4288*
STKQSR  016754          4108#  4121   4177   4288
STKS    001144          1272#  1793*  3430*  3472*  4104   4126*  4157*  4159   4165*  4187   4203*  4213   4225*
                        4245*
STKSRV  017026          2490   2758   2759   4123   4136#
STN   = 000001          1003#  1014
STNPWR  020734          4544   4545   4565#
```

```
$TOTAL= 000050          2626    2676    2678*   2965    5313#   5314
$TPB    001152          1275#   3943*   3954
$TPFLG  001157          1279#   1718*   3901    3954
$TPS    001150          1274#   3941    3954
$TRAP   021020          1682    4595#
$TRAP2  021042          4606#   4617
$TRK  = 000013          2026*   2338*   2450    2452*   3277    3559*   3646*   3761*   5302#   5303
$TRP  = 000014          4610#   4619#   4620#   4621#   4622#   4623#   4624    4625#   4626    4627#   4628#   4629#   4630#
                        4631#
$TRPAD  021054          4600    4617#
$TSTNM  001102          1252#   3864    3883
$TTYIN  020132          4299    4300    4312    4330    4344    4348#
$TYPBN= ****** U        4623
$TYPDS  016522          4045#   4622
$TYPE   016054          3901#   4610    4618
$TYPEC  016224          3922    3929    3936    3941#   3942    4247
$TYPEX  016272          3947    3949    3952#
$TYPOC  016320          3985#   4619
$TYPON  016334          3984    3987#   4621
$TYPOS  016274          3980#   4620
$WRDL = 000020          2031*   2152    2163    2165*   2166    2167*   2188    2197    2202    2210    2247    2252    2369*
                        2387    2460*   3253    3274    3562    3566*   3649    3653*   3712    5305#   5306
$WRDM = 000004          2032*   2033*   2387*   2388*   2461*   2462*   3567*   3654*   5298#   5299
$OFILL  016517          3981*   3985*   3995    4030#
$40CAT= ****** U        3874
.     = 034040          1218#   1222#   1230    1231#   1233#   1235#   1239#   1249#   1296    1677    2063    2072    2092
                        2205    2244    2246    2248    2250    3481    3484    3487    3490    3493    3496    3499    3502
                        3505    3508    3883    3954    4099#   4104    4108#   4109    4110#   4348#   4349    4356#   4585#
                        5340    5380    5420    5460    5500    5540    5580    5620    5658#   5727#   6336#   6617
```

```
A          2635#  2636   2645   2654   2663   2672
CKCHR      1003#  3121   3137
CKNUM      1003#  1779
COMMEN        1#  1143#
ENDCOM        1#  1143#
ERROR      1037#  1804   3246   3261   3394   3442   3447   3453   3458   3464   3470   3510   3515   3521   3527
           3533   3539   3556   3579   3582   3586   3592   3594   3601   3625   3631   3639   3641   3668   3672
           3797   5045
ESCAPE        1#  1143#
GETPRI        1#  1143#  4383   6646
GETSWR        1#  1003#  1143#  1708
MORETA     1243#  1297
MOVREG     1003#  3841
MULT          1#  1143#
NEWTST        1#  1143#
POP           1#  1143#  2937   3371   4086   4434   4479   4517
PUSH          1#  1143#  2922   3363   4045   4386   4459   4498
REPORT        1#  1143#
SCOPE      1038#
SETPRI        1#  1143#  4277
SETTRA     4610#  4619   4620   4621   4622   4624   4626   4627   4628   4629   4630
SETUP         1#  1143#  1672
SKIP          1#  1143#
SLASH         1#  1143#
SPACE      1143#
STARS         1#  1143#  1228   1245   1296   1322   1326   1353   1357   1369   1373   1659   1666   1944   1948
           2138   2142   2284   2288   2481   2485   3222   3226   3822   3828   3886   3957   4035   4103   4180
           4195   4266   4290   4360   4443   4489   4526   4589   4632   4636   5287   5291   5660   5664   5729
           5733   6026   6030   6634
SWRSU         1#  1143#  1686#
TRNTRP     4610#
TYPBIN        1#  1143#
TYPDEC        1#  1143#  3192
TYPNAM        1#  1143#
TYPNUM        1#  1143#
TYPOCS        1#  1143#
TYPOCT        1#  1143#  3189   4208
TYPTXT        1#  1143#
$$CMRE     1243#  1282   1283   1284   1285   1286   1287   1288   1289   1290   1291
$$CMTM     1243#
$$ESCA        1#  1143#
$$NEWT        1#  1143#
$$SET      4610#  4619   4620   4621   4622   4624   4626   4627   4628   4629   4630
$$SKIP        1#  1143#
.EQUAT        1#  1003#  1033
.HEADE        1#  1003#  1004
.KT11         1#
.SETUP        1#  1003#  1659
.SWRHI        1#  1003#  1014
.SWRLO     1003#  1021#  1022   1023   1024   1026   1029   1030   1031
.$ACTI        1#  1003#  1226
.$APTB        1#
.$APTH        1#
.$APTY        1#
.$ASTA        1#
.$CATC        1#  1003#  1216
```

```
.SCMTA      1#   1003#   1243
.SD82D      1#   1003#   4524
.SD820      1#
.SDIV       1#   1003#   4358
.SEOP       1#
.SERRO      1#   1003#   3826
.SERRT      1#   1003#
.SMULT      1#
.SPOWE      1#
.SRAND      1#   1003#   4487
.SRODE      1#
.SROOC      1#
.SRE2D      1#   1003#   4101
.SR2AZ      1#
.SSAVE      1#   1003#   4441
.SS82D      1#
.SS820      1#
.SSCOP      1#
.SSIZE      1#   1003#   6632
.SSUPR      1#
.STRAP      1#   1003#   4587
.STYPB      1#
.STYPD      1#   1003#   4033
.STYPE      1#   1003#   3884
.STYPO      1#   1003#   3955
.$40CA      1#
.1170       1#
```

```
ADC     2129  2133  2277  2281  4510  4513  4554
ADD     1963  1974  1981  2005  2019  2035  2075  2128  2132  2156  2166  2174  2190  2193  2210
        2216  2219  2225  2228  2276  2280  2317  2327  2337  2347  2363  2376  2386  2496  2530
        2539  2552  2559  2562  2564  2578  2604  2609  2614  2617  2840  2927  2932  3009  3058
        3132  3146  3165  3212  3248  3255  3263  3275  3396  3414  3416  3690  3691  3713  3723
        3795  3911  3983  3993  4065  4222  4232  4241  4412  4509  4511  4512  4514  4553  4555
        5209  5283
ASL     2165  2189  2215  2424  2827  2857  2880  2948  3023  3126  3128  3130  3162  3163  3164
        3254  3719  4255  4256  4257  4505  4599  4897  4920  4946  4992  5058  5064  5140  5164
        5182
ASLB    2030  4070  5055
ASR     2125  2167  2835  2859  3616  3617  3618  3619  3620  3621  4900  4948  4994  5060  5067
        5142  5166  5184
BCC     4071  4413
BCS     3127  3129  3131  3188
BEQ     1760  1768  1806  1813  1870  1882  1892  1901  1915  1919  1951  1971  1973  1978  1980
        2018  2045  2050  2057  2067  2102  2104  2124  2127  2131  2150  2155  2164  2170  2172
        2187  2192  2218  2240  2259  2279  2296  2300  2324  2334  2344  2356  2371  2380  2404
        2410  2495  2717  2727  2766  2804  2825  2833  2837  2851  2855  2861  2865  2876  2879
        3096  3167  3173  3178  3182  3196  3203  3245  3260  3269  3302  3304  3307  3312  3323
        3362  3377  3381  3393  3407  3423  3429  3479  3484  3487  3490  3493  3496  3499  3502
        3505  3508  3546  3555  3623  3638  3697  3704  3708  3710  3718  3726  3731  3741  3755
        3771  3784  3789  3791  3863  3866  3914  3949  4010  4147  4200  4239  4254  4282  4318
        4430  4806  4854  4857  4860  4863  4868  4978  5024  5047  5073  5077  5095  5144  5172
        5201  5231  5237  5239  5241  5243  5245  5266  5282
BGE     4425
BGT     2310  4017  4079  4171  4251  4401  4962  5012  5114  5120  5155  5268
BHI     1820  2449  2768  2966  2975  2981  3079  3288
BHIS    1747  1826  1829  1833  1837  1841  1845  1849  1853  2205  2244  2248  2362  2378  2641
        2650  2659  2668  2677  2978  3552  3564  3651
BIC     2316  2368  2405  2445  2504  2751  3081  3610  4007  4137  4162  4172  4190  4217  4252
        4385  4780  5005
BICB    2771  2856  2947  3022  4816  4991  5068  5083  5096  5149  5185  5186
BIS     4012  4013  4073  4074  4259  4402  4420  4557  4943  5018  5085  5088  5091  5093  5157
        5159
BISB    2023  4818  4896  4918  4919  4941  4942  4945
BIT     1871  1952  2295  2305  2437  2439  2456  2786  2963  2989  3157  3235  3242  3303  3380
        3422  3428  3432  3434  3436  3438  3440  3483  3486  3489  3492  3495  3498  3501  3504
        3507  3613  3637  3696  3703  3705  3709  3725  3740  3788  3790  3865  3872  4803  4805
        4807  4809  4814  5052  5062  5070  5072  5074  5167  5169  5171  5173  5226
BITB    1891  2126  2239  2366  2379  2381  2409  2549  2557  2808  2818  2854  2878  3270  3547
        3937  5025  5076
BLE     3200  3328  5213
BLO     2448  2770  2976  2982  2984  2992  3077  3331  3335  4313
BLOS    1757  1822  2153  2451  4301  4768  4772
BLT     3928  4018  4062  4078  4169  4249  4400  4550  4963  5112  5156
BMI     1883  2435  2464  2826  3785  4069  5001  5013  5148
BNE     1677  1693  1710  1712  1722  1735  1749  1754  1816  1872  1921  1931  1934  1936  1953
        1962  1965  1967  2004  2008  2042  2048  2063  2072  2085  2092  2115  2178  2195  2221
        2232  2242  2261  2298  2306  2313  2367  2373  2382  2408  2421  2431  2433  2438  2440
        2457  2550  2554  2558  2567  2624  2685  2695  2700  2704  2709  2713  2719  2729  2753
        2755  2773  2777  2781  2785  2787  2791  2809  2819  2821  2889  2892  2931  2936  2964
        2973  2990  3014  3049  3054  3098  3158  3205  3211  3216  3236  3243  3258  3271  3300
        3309  3317  3357  3379  3411  3413  3433  3435  3437  3439  3441  3481  3548  3614  3659
        3689  3706  3736  3794  3873  3908  3916  3924  3938  3945  4008  4067  4139  4145  4151
        4156  4164  4176  4186  4192  4220  4224  4230  4237  4244  4287  4305  4307  4323  4327
```

```
        4337  4417  4508  4561  4781  4804  4808  4810  4815  4852  4866  4882  4904  4926  4953
        4955  5026  5042  5053  5056  5063  5071  5075  5106  5123  5146  5168  5170  5174  5205
        5227  5233
BPL     2506  2511  3231  3878  3902  3942  4006  4053  4083  4160  4188  4214  4394  4419  4785
        5003
BR      1669  1695  1714  1751  1769  1787  1789  1792  1802  1885  1888  1903  1975  1982  1995
        2009  2020  2036  2052  2065  2069  2077  2096  2157  2168  2179  2200  2223  2233  2246
        2250  2264  2365  2384  2406  2423  2436  2466  2497  2513  2529  2538  2555  2560  2722
        2757  2775  2779  2783  2789  2793  2839  2841  2843  2863  2870  2883  2886  2893  2979
        2999  3018  3040  3042  3044  3052  3082  3120  3134  3191  3198  3213  3218  3290  3294
        3296  3310  3315  3321  3325  3415  3553  3575  3578  3581  3585  3591  3593  3630  3640
        3664  3667  3671  3721  3738  3787  3904  3921  3931  3940  3947  3984  3999  4020  4064
        4081  4154  4233  4260  4262  4316  4325  4331  4333  4404  4406  4409  4552  4812  4817
        4839  4892  4914  4935  4965  4968  4980  5007  5015  5017  5019  5030  5084  5087  5090
        5092  5098  5116  5121  5158  5163  5176  5179  5229  5235  5247  5250  5269  5284  6652
CLC     1744  4408
CLR     1668  1675  1706  1717  1720  1729  1730  1791  1793  1794  1818  1873  1910  1917  1954
        1992  1993  2000  2001  2002  2022  2043  2054  2059  2082  2089  2253  2358  2374  2411
        2502  2508  2521  2523  2533  2546  2575  2800  2801  2849  2874  2929  2934  2945  3012
        3020  3030  3045  3118  3136  3159  3179  3284  3285  3286  3332  3336  3369  3370  3568
        3629  3655  3693  3714  3997  4056  4059  4120  4157  4203  4211  4212  4276  4277  4298
        4321  4390  4421  4546  4766  4777  4783  4837  4889  4910  4989  4996  5031  5050  5065
        5109  5110  5162  5183  5202
CLRB    1718  1728  2730  3283  3318  3781  3920  3946  4085  4328  4338  4427  4562  4811  4843
        4984  5089  5127  5181
CMP     1676  1692  1711  1721  1746  1756  1819  1821  1825  1828  1832  1836  1840  1844  1848
        1852  1933  1935  1972  1979  2047  2130  2152  2191  2204  2217  2243  2247  2278  2309
        2361  2377  2407  2447  2640  2649  2658  2667  2676  2716  2752  2754  2965  2974  2977
        2980  2983  2991  3257  3287  3299  3330  3334  3410  3478  3480  3551  3554  3562  3622
        3649  3730  4077  4138  4144  4146  4150  4155  4163  4168  4170  4175  4185  4191  4219
        4229  4236  4248  4250  4286  4300  4312  4556  4767  4771  5119  5204  5212
CMPB    1920  2241  2297  2312  2370  2372  2450  2699  2703  2708  2712  2718  2728  2765  2767
        2769  2772  2776  2780  2784  2790  2803  2850  2864  2875  3048  3076  3078  3097  3268
        3545  3658  3707  3793  3913  3915  3923  3944  3948  4199  4223  4243  4304  4322  4326
        4336  5145  5230  5232  5236  5238  5240  5242  5244
COM     3183  3204
DEC     1900  1964  2007  2066  2154  2169  2171  2177  2194  2220  2226  2231  2258  2260  2553
        2566  2623  2836  2860  2891  2930  3013  3055  3161  3195  3199  3306  3308  3378  3412
        3688  3735  4283  4311  4416  4428  4559  4784  5267
DECB    2505  2510  3927  3930  4005  4016  4395
EMT     1037
HALT    1222  2245  2249  2274  3698  3742  3753  3769  3879  3903
INC     1734  1869  1902  1922  2006  2198  2199  2203  2325  2335  2345  2353  2551  2565  2642
        2651  2660  2669  2678  2694  2763  2838  2862  2890  3017  3051  3101  3125  3280  3329
        3333  3337  3368  3374  3868  4011  4019  4063  4167  4174  4258  4285  4405  4507  4551
        4779  5054  5117
INCB    2698  2702  2706  2707  2711  2715  2721  3786  3792  3862  3950
IOT     1038
JMP     1241  1242  1807  1824  1831  1835  1839  1843  1847  1851  1855  1864  1926  1932  1942
        1968  2086  2118  2302  2314  2459  2490  2759  2811  2814  2817  2900  2906  2967  3232
        3338  3424  3449  3460  3466  3474  3482  3485  3488  3491  3494  3497  3500  3503  3506
        3509  4143  4226
JSR     1707  1736  1779  1808  1905  1907  1924  1938  1955  1985  1991  1994  2011  2051  2053
        2055  2058  2060  2076  2083  2087  2088  2095  2107  2108  2109  2110  2111  2117  2272
        2293  2304  2308  2311  2326  2336  2346  2354  2360  2364  2392  2401  2419  2422  2429
        2442  2455  2468  2476  2556  2563  2574  2579  2586  2597  2598  2601  2602  2606  2607
```

	2611	2612	2620	2621	2631	2632	2725	2774	2778	2782	2788	2792	2810	2813	2823
	2830	2867	2882	2885	2894	3005	3032	3119	3121	3133	3138	3169	3233	3234	3237
	3247	3262	3293	3295	3305	3314	3324	3326	3359	3367	3373	3387	3395	3431	3448
	3459	3465	3471	3473	3516	3522	3528	3534	3540	3558	3565	3569	3573	3574	3576
	3577	3580	3583	3584	3587	3590	3602	3603	3604	3612	3626	3627	3632	3642	3643
	3645	3652	3656	3660	3663	3665	3666	3669	3670	3673	3674	3682	3687	3716	3751
	3767	3782	3817	3845	3874	3922	3929	3936	4141	4202	4247	4778	4838	4840	4855
	4858	4861	4864	4877	4880	4891	4902	4913	4924	4934	4951	4960	4967	4979	4981
	4982	5004	5006	5008	5010	5029	5080	5097	5115	5124	5125	5151	5152	5187	5249
MOV	1670	1674	1678	1680	1681	1682	1683	1684	1685	1688	1689	1690	1691	1696	1698
	1699	1700	1702	1703	1705	1719	1723	1725	1726	1727	1737	1738	1739	1740	1741
	1742	1755	1758	1764	1766	1767	1773	1774	1778	1785	1788	1790	1795	1799	1801
	1803	1805	1809	1810	1814	1817	1823	1827	1830	1834	1838	1842	1846	1850	1854
	1859	1874	1877	1884	1886	1887	1889	1890	1893	1906	1908	1909	1916	1923	1930
	1959	1960	1969	1976	1983	1999	2013	2021	2024	2027	2031	2032	2034	2046	2056
	2061	2064	2070	2073	2074	2081	2084	2090	2093	2094	2100	2101	2106	2112	2113
	2116	2122	2146	2147	2148	2149	2151	2158	2159	2160	2162	2173	2175	2176	2185
	2186	2188	2196	2197	2201	2202	2209	2211	2212	2213	2214	2224	2227	2229	2230
	2251	2252	2255	2256	2257	2262	2263	2270	2271	2275	2292	2294	2307	2315	2322
	2332	2342	2348	2352	2357	2359	2369	2387	2394	2395	2400	2418	2443	2444	2446
	2454	2460	2461	2470	2489	2494	2507	2512	2519	2520	2524	2525	2526	2527	2528
	2531	2534	2535	2536	2537	2540	2547	2548	2561	2577	2592	2596	2600	2605	2610
	2615	2616	2618	2625	2626	2693	2696	2723	2724	2731	2732	2748	2749	2750	2756
	2758	2761	2762	2802	2806	2807	2816	2828	2829	2831	2842	2844	2852	2858	2866
	2869	2881	2884	2887	2899	2905	2912	2923	2924	2925	2926	2928	2933	2937	2938
	2939	2944	2949	2958	2987	3006	3007	3000	3010	3015	3016	3024	3029	3031	3033
	3039	3041	3043	3046	3047	3057	3061	3062	3102	3117	3135	3147	3148	3166	3172
	3177	3180	3181	3189	3193	3207	3253	3256	3273	3274	3276	3279	3281	3289	3291
	3297	3298	3319	3320	3363	3364	3365	3366	3371	3372	3382	3384	3385	3402	3403
	3404	3406	3409	3417	3430	3472	3549	3561	3566	3567	3570	3571	3588	3599	3600
	3615	3624	3628	3648	3653	3654	3657	3661	3680	3683	3684	3685	3695	3711	3712
	3715	3727	3728	3733	3764	3765	3772	3815	3816	3818	3841	3842	3848	3849	3850
	3851	3852	3853	3854	3855	3856	3857	3858	3659	3860	3861	3864	3869	3905	3906
	3910	3925	3980	3988	3989	3990	3996	4003	4021	4022	4023	4024	4025	4046	4047
	4048	4049	4050	4051	4052	4057	4060	4080	4086	4087	4088	4089	4090	4092	4093
	4121	4122	4123	4124	4126	4165	4177	4208	4225	4240	4245	4274	4275	4278	4288
	4297	4299	4310	4341	4342	4343	4344	4386	4387	4388	4389	4391	4392	4393	4399
	4403	4411	4414	4422	4432	4433	4434	4435	4436	4437	4438	4459	4460	4461	4462
	4463	4464	4465	4466	4467	4468	4475	4476	4477	4478	4479	4480	4481	4482	4483
	4484	4499	4500	4501	4502	4503	4504	4515	4516	4517	4518	4519	4538	4539	4540
	4541	4542	4543	4544	4545	4595	4596	4600	4606	4607	4762	4763	4764	4765	4769
	4770	4773	4774	4775	4776	4782	4786	4832	4833	4835	4836	4842	4844	4850	4876
	4878	4879	4888	4893	4894	4898	4899	4901	4912	4915	4916	4921	4922	4923	4933
	4936	4937	4938	4939	4940	4944	4947	4949	4950	4964	4966	4976	4993	4995	4997
	5009	5014	5016	5027	5028	5033	5034	5036	5043	5044	5051	5059	5066	5078	5079
	5082	5128	5138	5141	5150	5153	5165	5175	5177	5180	5198	5200	5203	5208	5210
	5225	5228	5234	5246	5248	5252	5264	5270	5281	6640	6641	6642	6643	6644	6647
	6648	6649	6654	6655	6656	6657	6658	6659							
MOVB	1715	1987	2025	2026	2028	2029	2254	2301	2303	2318	2328	2338	2375	2383	2385
	2393	2413	2441	2452	2453	2465	2467	2469	2477	2478	2503	2509	2514	2576	2701
	2705	2710	2714	2720	2805	2822	2834	2853	2877	2946	2954	2995	3001	3021	3050
	3056	3080	3160	3185	3186	3277	3278	3282	3313	3358	3360	3386	3559	3560	3572
	3589	3609	3611	3646	3647	3662	3681	3686	3694	3749	3750	3761	3762	3763	3766
	3843	3846	3847	3871	3907	3935	3943	3981	3982	3985	3986	3987	3991	3994	3995
	4014	4055	4058	4072	4075	4084	4136	4161	4173	4189	4204	4216	4284	4303	4308
	4314	4319	4334	4558	4598	4801	4802	4813	4834	4869	4875	4887	4890	4895	4909

```
            4911  4917  4931  4932  4974  4990  4998  4999  5032  5035  5048  5061  5069  5086  5107
            5139  5160  5161  5178  5199
NEG         1724  2033  2388  2462  2697  3992  4054  4396  4397  4407  4426  4431
NOP         1939  1940  1941  2206  5081
RESET       1671  1937
ROL         3998  4000  4001  4002  4004  4410  4415  4506
ROR         1745  3187
RTI         1697  2733  2796  3882  3912  4026  4094  4166  4178  4246  4279  4289  4345  4439  4469
            4485  4608  4985
RTS         2134  2161  2208  2266  2282  2396  2414  2425  2471  2479  2498  2515  2541  2569  2581
            2588  2634  2643  2652  2661  2670  2679  2689  2812  2815  2845  2868  2895  2918  2940
            2959  2968  3025  3035  3063  3084  3103  3149  3220  3238  3249  3264  3272  3339  3375
            3388  3397  3418  3443  3454  3511  3517  3523  3529  3535  3541  3595  3605  3633  3675
            3699  3743  3756  3774  3796  3799  3819  3952  4127  4520  4564  4601  4788  4820  4845
            4871  4883  4905  4927  4956  4970  5037  5049  5057  5129  5188  5215  5253  5273  5285
            6660
SBC         4398  4548
SUB         1743  1750  1752  2105  2123  2163  2323  2333  2343  2475  2627  2628  2629  2630  2935
            3292  3383  3405  3550  3557  3644  3720  3722  3844  3870  4061  4547  4549  5113  6653
TRAP        4384  4610  4619  4620  4621  4622  4624  4626  4627  4628  4629  4630  6646
TST         1709  1753  1759  1786  1800  1811  1815  1914  1950  1961  1966  1970  1977  2003  2017
            2041  2044  2062  2068  2071  2091  2103  2114  2222  2299  2355  2402  2420  2430  2432
            2458  2522  2532  2684  2820  2832  2972  3034  3083  3099  3100  3142  3143  3144  3145
            3202  3210  3215  3230  3244  3259  3301  3311  3327  3361  3376  3392  3717  3754  3770
            3773  3783  3798  3877  3909  3917  3939  4009  4066  4076  4125  4142  4153  4158  4201
            4238  4253  4281  4306  4317  4340  4418  4423  4424  4429  4597  4841  4851  4853  4856
            4859  4862  4865  4881  4903  4925  4952  4954  4969  4977  5000  5002  5023  5046  5094
            5105  5111  5118  5122  5143  5206  5207  5211  5251  5265  5271  5272  6650  6651
TSTB        1748  1881  1918  2049  2434  2463  2726  2824  2888  3053  3095  3316  3322  3356  3901
            3941  4068  4082  4159  4187  4213  4867  4961  5011  5041  5147  5154
.ASCII      1293  1294  6197  6209  6226  6255  6451  6462  6463  6464  6487  6492  6619
.ASCIZ      1292  1295  4349  4350  4351  4352  4354  6032  6039  6044  6051  6059  6064  6073  6082
            6090  6096  6100  6104  6108  6113  6117  6123  6126  6133  6141  6145  6155  6166  6172
            6179  6185  6190  6191  6194  6200  6206  6213  6217  6221  6230  6240  6248  6259  6267
            6270  6277  6292  6290  6299  6308  6310  6312  6320  6328  6331  6459  6465  6466  6467
            6468  6470  6472  6474  6478  6482  6484  6498  6503  6509  6511  6514  6517  6521  6523
            6524  6527  6530  6573  6577  6582  6588  6590  6592  6594  6595  6596  6597  6598  6599
            6600  6602  6604  6606  6608  6610  6623
.BLKB        4108  4348  4585
.BLKW        4099  5658
.BYTE       1252  1253  1258  1259  1267  1268  1276  1277  1278  1279  1879  1989  2015  2594  2735
            2736  2737  2738  2739  2740  2914  2956  2997  3003  3341  3342  3801  4027  4028  4029
            4030  4346  4347  4656  4657  4658  4659  4660  4661  4662  4663  4690  4696  4741  4742
            4743  4744  4745  4746  4747  4748  5334  5374  5414  5454  5494  5534  5574  5614  5720
            5721  5722  5723  5724  6376  6377  6381  6382  6385  6386  6388  6389  6392  6393  6395
            6396  6399  6400  6403  6404  6407  6408  6411  6412  6415  6416  6418  6419  6422  6423
            6425  6426  6429  6430  6433  6434  6437  6438  6440  6441  6444  6445  6447  6448
.DSABL       4263
.ENABL          1  1003  4104
.END        6663
.ENDC       1009  1020  1021  1037  1129  1143  1225  1229  1233  1235  1246  1250  1252  1280  1292
            1293  1297  1323  1327  1354  1358  1370  1374  1383  1659  1660  1667  1678  1679  1680
            1682  1684  1686  1702  1711  1717  1945  1949  2139  2143  2285  2289  2482  2486  3223
            3227  3823  3829  3832  3862  3869  3874  3875  3876  3877  3882  3883  3887  3907  3958
            4036  4104  4135  4144  4148  4179  4181  4196  4227  4263  4267  4278  4290  4291  4299
            4301  4304  4332  4349  4350  4356  4361  4385  4444  4490  4527  4590  4596  4599  4618
```

```
                4619  4620  4621  4622  4623  4624  4625  4626  4627  4628  4629  4630  4631  4633  4637
                5288  5292  5661  5665  5730  5734  6027  6031  6635  6648  6662
.EQUIV          1037  1038  1046  1061  1062  1091  1092  1093  1094  1095  1096  1097  1098  1099  1100
                1119  1120  1121  1122  1123  1124  1125  1126  1127  1128
.EVEN           2741  4110  4356  5727  6336  6539  6613  6630
.IF             1005  1020  1021  1035  1101  1129  1225  1228  1231  1233  1245  1249  1251  1280  1292
                1296  1297  1322  1326  1353  1357  1369  1373  1659  1666  1673  1678  1680  1682  1684
                1686  1702  1708  1711  1944  1948  2138  2142  2284  2288  2481  2485  3222  3226  3822
                3828  3831  3841  3865  3872  3874  3875  3877  3881  3882  3883  3886  3907  3957  4035
                4103  4105  4133  4138  4144  4180  4181  4195  4219  4266  4267  4277  4290  4298  4300
                4304  4305  4348  4349  4356  4360  4385  4443  4489  4526  4589  4595  4599  4610  4619
                4620  4621  4622  4623  4624  4626  4627  4628  4629  4630  4631  4632  4636  5287  5291
                5660  5664  5729  5733  6026  6030  6634  6638  6647  6648
.IFF             1020  1021  1035  1229  1233  1246  1249  1251  1280  1297  1323  1327  1354  1358  1370
                1374  1660  1667  1678  1945  1949  2139  2143  2285  2289  2482  2486  3223  3227  3823
                3829  3831  3865  3881  3882  3883  3887  3958  4036  4104  4181  4196  4206  4267  4270
                4278  4290  4291  4300  4332  4348  4361  4444  4490  4527  4590  4596  4633  4637  5288
                5292  5661  5665  5730  5734  6027  6031  6635  6638  6648
.IFT            3875  4269  4274  6642  6648  6658  6662
.IFTF           3874  4206  4267  4270  6639  6642  6654  6658
.IIF            1004  1009  1014  1018  1019  1020  1021  1022  1023  1024  1025  1026  1027  1029  1030
                1031  1222  1296  1679  1686  3190  3194  3832  3833  3834  3835  3836  3840  3880  3882
                3883  3954  4104  4110  4149  4150  4209  4340  4349  4356  4618  4619  4620  4621  4622
                4624  4626  4627  4628  4629  4630
.IRP            1297  1659  2600  2610  2627  2923  2937  3363  3371  3841  3850  4046  4086  4386  4434
                4459  4479  4499  4517  5331  5711
.LIST              1  1003  1143  1222  1280  1282  1283  1284  1285  1286  1287  1288  1289  1290  1291
                1292  1659  1686  2635  3882  4290  4610  4618  4619  4620  4621  4622  4623  4624  4625
                4626  4627  4628  4629  4630  4631  6632
.MACRO             1  1003  1021  1243  2635  4610
.MCALL          1003  1143  1686
.NLIST             1  1003  1143  1222  1280  1282  1283  1284  1285  1286  1287  1288  1289  1290  1291
                1292  1659  1686  2635  3882  4290  4610  4618  4619  4620  4621  4622  4623  4624  4625
                4626  4627  4628  4629  4630  4631  6632
.PAGE           1243  1383
.REM               1
.REPT           1222  1282  3616  5335  5341  5375  5381  5415  5421  5455  5461  5495  5501  5535  5541
                5575  5581  5615  5621  5689  5755  5891
.SBTTL          1014  1033  1216  1226  1237  1243  1324  1355  1371  1383  1661  1672  1708  1946  2140
                2286  2483  3224  3824  3826  3884  3955  4033  4101  4358  4441  4487  4524  4587  4610
                4634  5289  5662  5731  6028  6632
.TITLE          1004
.WORD           1222  1223  1224  1234  1251  1254  1255  1256  1257  1260  1261  1262  1263  1264  1265
                1266  1269  1270  1271  1280  1282  1283  1284  1285  1286  1287  1288  1289  1290  1291
                1297  1298  1299  1300  1301  1302  1303  1304  1305  1306  1307  1308  1309  1310  1311
                1312  1313  1315  1316  1317  1318  1319  1320  1328  1329  1330  1331  1332  1333  1336
                1337  1339  1340  1341  1350  1359  1363  1364  1365  1366  1367  1375  1376  1378  1772
                1862  1895  1898  2136  2743  2916  3060  3065  3171  3176  3209  3344  3345  3346  3347
                3348  3349  3350  3351  3352  3802  3820  3951  4031  4105  4106  4107  4521  4522  4560
                4617  4642  4643  4649  4670  4676  4684  4704  4708  4711  4714  4717  4720  4728  4729
                4730  4731  4732  4733  4734  4735  5335  5336  5337  5338  5339  5340  5341  5342  5343
                5344  5345  5346  5347  5348  5349  5350  5351  5352  5353  5354  5355  5356  5357  5358
                5359  5360  5361  5362  5363  5364  5365  5366  5367  5368  5369  5370  5375  5376  5377
                5378  5379  5380  5381  5382  5383  5384  5385  5386  5387  5388  5389  5390  5391  5392
                5393  5394  5395  5396  5397  5398  5399  5400  5401  5402  5403  5404  5405  5406  5407
                5408  5409  5410  5415  5416  5417  5418  5419  5420  5421  5422  5423  5424  5425  5426
```

5427	5428	5429	5430	5431	5432	5433	5434	5435	5436	5437	5438	5439	5440	5441
5442	5443	5444	5445	5446	5447	5448	5449	5450	5455	5456	5457	5458	5459	5460
5461	5462	5463	5464	5465	5466	5467	5468	5469	5470	5471	5472	5473	5474	5475
5476	5477	5478	5479	5480	5481	5482	5483	5484	5485	5486	5487	5488	5489	5490
5495	5496	5497	5498	5499	5500	5501	5502	5503	5504	5505	5506	5507	5508	5509
5510	5511	5512	5513	5514	5515	5516	5517	5518	5519	5520	5521	5522	5523	5524
5525	5526	5527	5528	5529	5530	5535	5536	5537	5538	5539	5540	5541	5542	5543
5544	5545	5546	5547	5548	5549	5550	5551	5552	5553	5554	5555	5556	5557	5558
5559	5560	5561	5562	5563	5564	5565	5566	5567	5568	5569	5570	5575	5576	5577
5578	5579	5580	5581	5582	5583	5584	5585	5586	5587	5588	5589	5590	5591	5592
5593	5594	5595	5596	5597	5598	5599	5600	5601	5602	5603	5604	5605	5606	5607
5608	5609	5610	5615	5616	5617	5618	5619	5620	5621	5622	5623	5624	5625	5626
5627	5628	5629	5630	5631	5632	5633	5634	5635	5636	5637	5638	5639	5640	5641
5642	5643	5644	5645	5646	5647	5648	5649	5650	5654	5656	5666	5668	5672	5676
5680	5684	5688	5689	5690	5691	5692	5693	5694	5695	5696	5697	5698	5699	5700
5701	5702	5703	5704	5705	5706	5707	5708	5711	5712	5713	5714	5715	5716	5717
5718	5735	5736	5737	5738	5739	5740	5741	5742	5743	5744	5745	5746	5747	5748
5749	5750	5751	5752	5754	5755	5756	5757	5758	5759	5760	5761	5762	5763	5764
5765	5766	5767	5768	5769	5771	5772	5773	5774	5775	5776	5777	5778	5779	5780
5781	5782	5783	5784	5785	5786	5788	5789	5790	5791	5792	5793	5794	5795	5796
5797	5798	5799	5800	5801	5802	5803	5805	5806	5807	5808	5809	5810	5811	5812
5813	5814	5815	5816	5817	5818	5819	5820	5822	5823	5824	5825	5826	5827	5828
5829	5830	5831	5832	5833	5834	5835	5836	5837	5839	5840	5841	5842	5843	5844
5845	5846	5847	5848	5849	5850	5851	5852	5853	5854	5856	5857	5858	5859	5860
5861	5862	5863	5864	5865	5866	5867	5868	5869	5870	5871	5873	5874	5875	5876
5877	5878	5879	5880	5881	5882	5883	5884	5885	5886	5887	5888	5890	5891	5892
5893	5894	5895	5896	5897	5898	5899	5900	5901	5902	5903	5904	5905	5907	5908
5909	5910	5911	5912	5913	5914	5915	5916	5917	5918	5919	5920	5921	5922	5924
5925	5926	5927	5928	5929	5930	5931	5932	5933	5934	5935	5936	5937	5938	5939
5941	5942	5943	5944	5945	5946	5947	5948	5949	5950	5951	5952	5953	5954	5955
5956	5958	5959	5960	5961	5962	5963	5964	5965	5966	5967	5968	5969	5970	5971
5972	5973	5975	5976	5977	5978	5979	5980	5981	5982	5983	5984	5985	5986	5987
5988	5989	5990	5992	5993	5994	5995	5996	5997	5998	5999	6000	6001	6002	6003
6004	6005	6006	6007	6009	6010	6011	6012	6013	6014	6015	6016	6017	6018	6019
6020	6021	6022	6023	6024	6338	6339	6340	6341	6342	6344	6345	6346	6348	6350
6351	6353	6355	6357	6358	6360	6362	6364	6367	6370	6372	6375	6380	6384	6387
6391	6394	6398	6402	6406	6410	6414	6417	6421	6424	6428	6432	6436	6439	6443
6446	6541	6542	6543	6544	6545	6546	6547	6548	6549	6550	6551	6552	6553	6554
6555	6556	6557	6558	6559	6560	6561	6562	6563	6564	6565	6566	6567	6568	6569
6570	6571	6615	6661											

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

#,NOW.SEQ/SOL/CRF/NL:TOC/PAGNUM=DZRP18.SML,DZRP18.CMB
RUN-TIME: 44 66 10 SECONDS
RUN-TIME RATIO: 284/121=2.3
CORE USED: 35K (69 PAGES)

Spooler routine 23 Seconds, 99 KCS, 664 disk reads, 3 disk writes, 157 pages