

RP04/05/06

READ/WRITE TEST
MD-11-DZRJA-A

EP-DZRJA-A-01-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
MADE IN USA

801

NO-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 19:27 PAGE 1
RPO456.01C 14-APR-76 00:00

1

CO1

NO-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 2
RPO456.010 14-APR-76 00:00

2

E01

MD-11-DZRJA-RA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 5
DZRJA.014 19-APR-76 00:00 TRAP CATCHER

```
59      ;#ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"  
60      ;#SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS  
61      ;#LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS  
62      ;=174  
63      000174 000174  
64      000174 000000  
65      000176 000000  
66      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER  
67      SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER  
68      .SBTTL ACT11 HOOKS  
69      ;*****  
70      ;HOOKS REQUIRED BY ACT11  
71      $SVPC=      ;SAVE PC  
72      =46  
73      SENDAD      ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP  
74      =52  
75      .WORD 0      ;;2)SET LOC.52 TO ZERO  
76      =$SVPC      ;; RESTORE PC  
77      .SBTTL STARTING ADDRESSES  
78      =200  
79      ;#200 = NORMAL START  
80      JMP      @#START1  
81      ;#204 = SELECT OPERATING PARAMETERS  
82      JMP      @#START2  
83      ;#210 = SELECT RH11/RPO4/5/6 ADDRESSES  
84      JMP      @#START3  
85      ;#214 = COMBINATION OF 204 AND 210  
86      JMP      @#START4  
87      .SBTTL BASIC DEFINITIONS  
88      ;#INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***  
89      STACK= 1100  
90      .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL  
91      .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL  
92      ;#MISCELLANEOUS DEFINITIONS  
93      HT= 11      ;;CODE FOR HORIZONTAL TAB  
94      LF= 12      ;;CODE FOR LINE FEED  
95      CR= 15      ;;CODE FOR CARRIAGE RETURN  
96      CRLF= 200      ;;CODE FOR CARRIAGE RETURN-LINE FEED  
97      PS= 177776      ;;PROCESSOR STATUS WORD  
98      .EQUIV PS,PSW  
99      STKLMT= 177774      ;;STACK LIMIT REGISTER  
100      PIRQ= 177772      ;;PROGRAM INTERRUPT REQUEST REGISTER  
101      DSMR= 177570      ;;HARDWARE SWITCH REGISTER  
102      DDISP= 177570      ;;HARDWARE DISPLAY REGISTER  
103      ;#GENERAL PURPOSE REGISTER DEFINITIONS  
104      R0= %0      ;;GENERAL REGISTER  
105      R1= %1      ;;GENERAL REGISTER  
106      R2= %2      ;;GENERAL REGISTER  
107      R3= %3      ;;GENERAL REGISTER  
108      R4= %4      ;;GENERAL REGISTER  
109      R5= %5      ;;GENERAL REGISTER  
110      R6= %6      ;;GENERAL REGISTER  
111  
112  
113  
114
```


115 000007
116 000006
117 000007
118
119
120 000000
121 000040
122 000100
123 000140
124 000200
125 000240
126 000300
127 000340
128
129
130 100000
131 040000
132 020000
133 010000
134 004000
135 002000
136 001000
137 000400
138 000200
139 000100
140 000040
141 000020
142 000010
143 000004
144 000002
145 000001
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170

R7= %7
SP= %6
PC= %7

:::GENERAL REGISTER
:::STACK POINTER
:::PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS
PR0= 0
PR1= 40
PR2= 100
PR3= 140
PR4= 200
PR5= 240
PR6= 300
PR7= 340

:::PRIORITY LEVEL 0
:::PRIORITY LEVEL 1
:::PRIORITY LEVEL 2
:::PRIORITY LEVEL 3
:::PRIORITY LEVEL 4
:::PRIORITY LEVEL 5
:::PRIORITY LEVEL 6
:::PRIORITY 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

BASIC DEFINITIONS

171 000004
 172 000002
 173 000001
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186 000004
 187 000010
 188 000014
 189 000014
 190 000014
 191 000020
 192 000024
 193 000030
 194 000034
 195 000060
 196 000064
 197 000240
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211 000100
 212 000200
 213 000400
 214 001000
 215 002000
 216 020000
 217 040000
 218
 219
 220
 221
 222
 223
 224
 225 000001
 226 000002
 227 000004
 228 000010

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

.SBTTL RH11 REGISTERS

;;*****

;CONTROL AND STATUS REGISTER 1 (RPCS1)

IE= 100 ;INTERRUPT ENABLE (BIT #6)
 RDY= 200 ;READY (BIT #7)
 A16= 400 ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
 A17= 1000 ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
 PSEL= 2000 ;PORT SELECT (BIT #10)
 MCPE= 20000 ;MASSBUSS PARITY ERROR (BIT #13)
 TRE= 40000 ;TRANSFER ERROR (BIT #14)
 ;SC= 100000 ;SPECIAL CONDITION (BIT #15)

;WORD COUNT REGISTER (RPWC)
 ;(EACH BIT IS CALLED BY BIT NUMBER)

;BUS ADDRESS REGISTER (RPBA)
 ;(EACH BIT IS CALLED BY BIT NUMBER)

;CONTROL AND STATUS REGISTER 2 (RPCS2)

US1= 1 ;UNIT SELECT (BIT #0)
 US2= 2 ;UNIT SELECT (BIT #1)
 US4= 4 ;UNIT SELECT (BIT #2)
 BAI= 10 ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)

227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282

000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

:PAT= 20
:CLR= 40
:IR= 100
:OR= 200
:MPE= 400
:MXF= 1000
:PGE= 2000
:NEH= 4000
:NED= 10000
:UPE= 20000
:WCE= 40000
:DLT= 100000

:MASSBUS PARITY TEST (BIT #4)
:CLEAR (BIT #5)
:INPUT READY (BIT #6)
:OUTPUT READY (BIT #7)
:MASS BUS PARITY ERROR (BIT #8)
:MISSED TRANSFER ERROR (BIT #9)
:PROGRAM ERROR (BIT #10)
:NON EXISTENT MEMORY (BIT #11)
:NON EXISTENT DRIVE (BIT #12)
:UNIBUS PARITY ERROR (BIT #13)
:WRITE CHECK ERROR (BIT #14)
:DATA LATE (BIT #15)

;DATA BUFFER REGISTER (RPDB)
;(EACH BIT IS CALLED BY BIT NUMBER)

::*****

.SBTTL RPO4/5/6 REGISTERS

::*****

;CONTROL AND STATUS 1 REGISTER. (#00)

000001
000002
000004
000010
000020
000040
004000

G0= 1
F1= 2
F2= 4
F3= 10
F4= 20
F5= 40
DVA= 4000

:GO BIT (BIT #0)
:FUNCTION CODE BIT #1
:FUNCTION CODE BIT #2
:FUNCTION CODE BIT #3
:FUNCTION CODE BIT #4
:FUNCTION CODE BIT #5
:DEVICE AVAILABLE (BIT #11)

;DRIVE STATUS REGISTER (RPDS1) (#01)

000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

:DF5= 1
:DF20= 2
:DIGB= 4
:GRV= 10
:DL64= 20
:DE1= 40
:VV= 100
:DRY= 200
:DPR= 400
:PGM= 1000
:LST= 2000
:WRL= 4000
:MOL= 10000
:PIP= 20000
:ERR= 40000
:ATA= 100000

DRIVE FORWARD 5"/SEC. (BIT #0)
:DRIVE FORWARD 20"/SEC. (BIT #1)
:DRIVE TO INNER GUARD BAND (BIT #2)
:GO REVERSE (BIT #3)
:DIFFERENCE LESS THAN 64 (BIT #4)
:DIFFERENCE EQUALS 1 (BIT #5)
:VOLUME VALID (BIT #6)
:DRIVE READY (BIT #7)
:DRIVE PRESENT (BIT #8)
:PROGRAMABLE (BIT #9)
:LAST SECTOR TRANSFERRED (BIT #10)
:WRITE LOCK (BIT #11)
:MEDIUM ON-LINE (BIT #12)
:POSITIONING OPERATION IN PROGRESS (BIT #13)
:COMPOSITE ERROR (BIT #14)
:ATTENTION ACTIVE (BIT #15)

;ERROR REGISTER #01 (RPER1) (#02)

000001
000002

ILF= 1
ILR= 2

:ILLEGAL FUNCTION (BIT #0)
:ILLEGAL REGISTER (BIT #1)

000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000
000001
000002
000004
000010
000020
000040
000200
000001
000002
000004
000010
000020
000040
000100
000200
000001
000002
000004
000010
000020
000040
000100
000200
000001
000002
000004
000010
000020
000040
000100
000200
000400
004000
020000
040000
100000

000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

RMR= 4
PAR= 10
FER= 20
MCF= 40
ECH= 100
HCE= 200
HCRC= 400
AOE= 1000
IAE= 2000
MLE= 4000
JTE= 10000
OPT= 20000
UNS= 40000
DCK= 100000

:REGISTER MODIFICATION REFUSED (BIT #2)
:PARITY ERROR (BIT #3)
:FORMAT ERROR (BIT #4)
:WRITE CLOCK FAIL (BIT #5)
:ECC HARD ERROR (BIT #6)
:HEADER COMPARE ERROR (BIT #7)
:HEADER CRC ERROR (BIT #8)
:ADDRESS OVERFLOW ERROR (BIT #9)
:INVALID ADDRESS ERROR (BIT #10)
:WRITE LOCK ERROR (BIT #11)
:DRIVE TIMING ERROR (BIT #12)
:OPERATION INCOMPLETE (BIT #13)
:DRIVE UNSAFE (BIT #14)
:DATA CHECK ERROR (BIT 15)

;MAINTAINABILITY REGISTER (RPMR) (#03)

000001
000002
000004
000010
000020
000040
000200

DMD= 1
MCLK= 2
MINX= 4
MSTCK= 10
MRD= 20
MWR= 40
DTSY= 200

:DIAGINOSTIC MODE (BIT #0)
:MAINTAINABILITY CLOCK (BIT #1)
:MAINTAINABILITY INDEX (BIT #2)
:MAINTAINABILITY SECTOR CLOCK (BIT #3)
:MAINTAINABILITY READ (BIT #4)
:MAINTAINABILITY WRITE (BIT #5)
:MAINTAINABILITY SYNC DETECTED (BIT #7)

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

000001
000002
000004
000010
000020
000040
000100
000200

AT0= 1
AT1= 2
AT2= 4
AT3= 10
AT4= 20
AT5= 40
AT6= 100
AT7= 200

:DEVICE 0 (BIT #0)
:DEVICE 1 (BIT #1)
:DEVICE 2 (BIT #2)
:DEVICE 3 (BIT #3)
:DEVICE 4 (BIT #4)
:DEVICE 5 (BIT #5)
:DEVICE 6 (BIT #6)
:DEVICE 7 (BIT #7)

;DESIRED SECTOR/TRACK ADDRESS REGISTER (RPDA) (#05)
;(EACH BIT IS CALLED BY BIT NUMBER)

;DRIVE TYPE REGISTER (RPDT) (#06)

000001
000002
000004
000010
000020
000040
000100
000200
000400
004000
020000
040000
100000

DT00= 1
DT01= 2
DT02= 4
DT03= 10
DT04= 20
DT05= 40
DT06= 100
DT07= 200
DT08= 400
DR0= 4000
M0H= 20000
TAP= 40000
NBA= 100000

:DRIVE TYPE NUMBER BIT 1
:DRIVE TYPE NUMBER BIT 2
:DRIVE TYPE NUMBER BIT 3
:DRIVE TYPE NUMBER BIT 4
:DRIVE TYPE NUMBER BIT 5
:DRIVE TYPE NUMBER BIT 6
:DRIVE TYPE NUMBER BIT 7
:DRIVE TYPE NUMBER BIT 8
:DRIVE TYPE NUMBER BIT 9
:DRIVE REQUEST REQUIRED (BIT #11)
:MOVING HEAD (BIT #13)
:TAPE DRIVE (BIT #14)
:NOT BLOCK ADDRESSED (BIT #15)

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

333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394

000001
000002
000004
000010
000020
000040
000100
000200

001000
002000
004000
010000
020000
040000
100000

EXT1= 1
EXT2= 2
EXT4= 4
EXT10= 10
EXT20= 20
EXT40= 40
SC1= 100
SC2= 200
SC4= 400
SC10= 1000
SC20= 2000
TRK1= 4000
TRK2= 10000
TRK4= 20000
TRK10= 40000
TRK20= 100000

:EXTENSION 1 (BIT #0)
:EXTENSION 2 (BIT #1)
:EXTENSION 3 (BIT #2)
:EXTENSION 4 (BIT #3)
:EXTENSION 5 (BIT #4)
:EXTENSION 6 (BIT #5)
:SECTOR COUNT FIELD 0 (BIT #6)
:SECTOR COUNT FIELD 1 (BIT #7)
:SECTOR COUNT FIELD 2 (BIT #8)
:SECTOR COUNT FIELD 3 (BIT #9)
:SECTOR COUNT FIELD 4 (BIT #10)
:TRACK FIELD 1 (BIT #11)
:TRACK FIELD 2 (BIT #12)
:TRACK FIELD 3 (BIT #13)
:TRACK FIELD 4 (BIT #14)
:TRACK FIELD 5 (BIT #15)

;RPO4 ERROR REGISTER #2 (RPER2) (#10)

000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
100000

WCU= 1
CSF= 2
WSU= 4
CSU= 10
MSE= 20
TDF= 40
TUF= 100
FEN= 200
WRU= 400
MHS= 1000
NHS= 2000
IXE= 4000
VU30= 10000
PLU= 20000
ACU= 100000

:WRITE CURRENT UNSAFE (BIT #0)
:CURRENT SINK FAILURE (BIT #1)
:WRITE SELECT UNSAFE (BIT #2)
:CURRENT SWITCH UNSAFE (BIT #3)
:MOTOR SEQUENCE ERROR (BIT #4)
:TRANSITIONS DETECTOR FAILURE (BIT #5)
:TRANSITIONS UNSAFE (BIT #6)
:FAILSAFE ENABLED (BIT #7)
:WRITE READY UNSAFE (BIT #8)
:MULTIPLE HEAD SELECT (BIT #9)
:NO HEAD SELECTION (BIT #10)
:INDEX ERROR (BIT #11)
:30VOLT UNSAFE (BIT #12)
:PLO UNSAFE (BIT #13)
:AC UNSAFE (BIT #15)

;RPO5/6 ERROR REGISTER #02 (RPER2) (#10)

000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
020000

WCU= 1
CSF= 2
WSU= 4
CSU= 10
RAW= 20
TDF= 40
TUF= 100
ABS= 200
WRU= 400
MHS= 1000
NHS= 2000
IXE= 4000
PLU= 20000

:WRITE CURRENT UNSAFE (BIT #0)
:CURRENT SINK FAILURE (BIT #1)
:WRITE SELECT UNSAFE (BIT #2)
:CURRENT SWITCH UNSAFE (BIT #3)
:READ AND WRITE (BIT #4)
:TRANSITIONS DETECTOR FAILURE (BIT #5)
:TRANSITIONS UNSAFE (BIT #6)
:ABNORMAL STOP (BIT #7)
:WRITE READY UNSAFE (BIT #8)
:MULTIPLE HEAD SELECT (BIT #9)
:NO HEAD SELECTION (BIT #10)
:INDEX ERROR (BIT #11)
:PLO UNSAFE (BIT #12)

;OFFSET REGISTER (RPOF) (#11)

000001
000002

OF25= 1
OF50= 2

:OFFSET 25 MICRO INCHES (BIT #0)
:OFFSET 50 MICRO INCHES (BIT #1)

L01

451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467

000123
000131
000151
000153
000161
000163
000171
000173
000141
000143
000145

PACK=123
SEARCH=131
WRCKD=151
WRCKHD=153
WRITE=161
MRTHD=163
READ=171
READHD=173
GETREG=141
SETFORM=143
SELDIV=145

;OTHER EQUATES

177400
010000

SCTRWC = -256.
FMT22=10000

:WORD COUNT FOR SECTOR
:FORMAT 22 BIT

.SBTTL COMMON TAGS

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

468
469
470
471
472
473
474 001100
475 001100 000000
476 001100 000000
477 001102 000
478 001103 000
479 001104 000000
480 001106 000000
481 001110 000000
482 001112 000000
483 001114 000
484 001115 001
485 001116 000000
486 001120 000000
487 001122 000000
488 001124 000000
489 001126 000000
490 001130 000000
491 001132 000000
492 001134 000
493 001135 000
494 001136 000000
495 001140 177570
496 001142 177570
497 001144 177560
498 001146 177562
499 001150 177564
500 001152 177566
501 001154 000
502 001155 002
503 001156 012
504 001157 000
505 001160 000000
506
507 001162 000000
508 001164 000000
509 001166 000000
510 001170 000000
511 001172 000000
512 001174 000000
513 001176 000000
514 001200 000000
515 001202 000000
516 001204 000000
517 001206 000000
518 001210 177607 000377
519 001214 077
520 001215 015
521 001216 000012
522
523 000015

.S1100
SCMTAG: .WORD 0
SPASS: .WORD 0
STSTNM: .BYTE 0
SERFLG: .BYTE 0
SICNT: .WORD 0
SLPADR: .WORD 0
SLPERR: .WORD 0
SERTTL: .WORD 0
SITEMB: .BYTE 0
SERMAX: .BYTE 1
SERRPC: .WORD 0
SGDADR: .WORD 0
SBDADR: .WORD 0
SGDDAT: .WORD 0
SBDAT: .WORD 0
SAUTOB: .BYTE 0
SINTAG: .BYTE 0
SWR: .WORD DSWR
DISPLAY: .WORD DDISP
STKS: 177560
STKB: 177562
STPS: 177564
STPB: 177566
SNLL: .BYTE 0
SFILLS: .BYTE 2
SFILLC: .BYTE 12
STPFLG: .BYTE 0
SREGAD: .WORD 0
SREG0: .WORD 0
SREG1: .WORD 0
SREG2: .WORD 0
SREG3: .WORD 0
SREG4: .WORD 0
SREG5: .WORD 0
STMP0: .WORD 0
STMP1: .WORD 0
STMP2: .WORD 0
STIMES: 0
SESCAPE: 0
SBELL: .ASCIZ <207><377><377>
SQUES: .ASCII ?/
SCRLF: .ASCII <15>
SLF: .ASCIZ <12>
CR = 15

START OF COMMON TAGS
CONTAINS PASS COUNT
CONTAINS THE TEST NUMBER
CONTAINS ERROR FLAG
CONTAINS SUBTEST ITERATION COUNT
CONTAINS SCOPE LOOP ADDRESS
CONTAINS SCOPE RETURN FOR ERRORS
CONTAINS TOTAL ERRORS DETECTED
CONTAINS ITEM CONTROL BYTE
CONTAINS MAX. ERRORS PER TEST
CONTAINS PC OF LAST ERROR INSTRUCTION
CONTAINS ADDRESS OF 'GOOD' DATA
CONTAINS ADDRESS OF 'BAD' DATA
CONTAINS 'GOOD' DATA
CONTAINS 'BAD' DATA
RESERVED--NOT TO BE USED
AUTOMATIC MODE INDICATOR
INTERRUPT MODE INDICATOR
ADDRESS OF SWITCH REGISTER
ADDRESS OF DISPLAY REGISTER
TTY KBD STATUS
TTY KBD BUFFER
TTY PRINTER STATUS REG. ADDRESS
TTY PRINTER BUFFER REG. ADDRESS
CONTAINS NULL CHARACTER FOR FILLS
CONTAINS # OF FILLER CHARACTERS REQUIRED
INSERT FILL CHARS. AFTER A "LINE FEED"
"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
CONTAINS THE ADDRESS FROM WHICH (SREG0) WAS OBTAINED
CONTAINS ((SREGAD)+0)
CONTAINS ((SREGAD)+2)
CONTAINS ((SREGAD)+4)
CONTAINS ((SREGAD)+6)
CONTAINS ((SREGAD)+10)
CONTAINS ((SREGAD)+12)
USER DEFINED
USER DEFINED
USER DEFINED
MAX. NUMBER OF ITERATIONS
ESCAPE ON ERROR ADDRESS
CODE FOR BELL
QUESTION MARK
CARRIAGE RETURN
LINE FEED

524		000012		LF	=	12		
525	001220	000000		C.SWR:	.WORD	0		: CONTROL SWITCHES
526	001222	000000		SAVCSW:	.WORD	0		: PREVIOUS CONTENTS OF 'C.SWR'
527	001224	000000		CNTRLC:	.WORD	0		: CONTROL "C" FLAG
528	001226	000000		BUSADR:	.WORD	0		: GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
529	001230	000000		LPTAVL:	.WORD	0		: LPT AVAILABLE STATUS (0=NO, 1=YES)
530	001232	000000		DRVSEL:	.WORD	0		: DRIVES SELECTED FOR TESTING
531	001234	037777	000000	TSTNMS:	.WORD	37777,0		: RUN TESTS 0-15
532	001240	000000	000000	OPNFLG:	.WORD	0,0		: MODIFY TEST PARAMETER FLAGS
533	001244	000000		CLKSTA:	.WORD	0		: CLOCK STATUS (0=NO CLOCK,+1=KW11-P, AND -1=KW11-L)
535	001246	000020		TICKMS:	.WORD	16.		: 16 MILLISECOND PER CLOCK TICK
536	001250	040432		TICKUS:	.WORD	16666.		: 16666 MICROSECONDS PER CLOCK TICK
537	001252	000000		BYPASS:	.WORD	0		
538	001254	000000		CHKDRV:	.WORD	0		: DRIVE UNDER TEST
539	001256	000000		DRVMSK:	.WORD	0		: DRIVE MASK BIT
540	001260	000000		SVSTAT:	.WORD	0		: STATUS/ERROR INDICATOR IS SAVED HERE ON AN ERROR
542	001262	000000		CYL.RD:	.WORD	0		: CYLINDER READ
543	001264	000000		TRK.RD:	.WORD	0		: TRACK READ
544	001266	000000		SEC.RD:	.WORD	0		: SECTOR READ
545	001270	000000		CYL.DS:	.WORD	0		: CYLINDER DESIRED
546	001272	000000		SEC.DS:	.WORD	0		: SECTOR DESIRED
547	001274	000000		TRK.DS:	.WORD	0		: TRACK DESIRED
548	001276	000000		TIM.UP:	.WORD	0		: MINIMUM TIME
549	001300	000000			.WORD	0		: NUMBER OF COUNTS BELOW MIN. LIMIT
550	001302	000000			.WORD	0		: MAXIMUM TIME
551	001304	000000			.WORD	0		: NUMBER OF COUNTS ABOVE MAX. LIMIT
552	001306	000000	000000		.WORD	0,0		: TOTAL TIME OF ALL SEEKS
553	001312	000000			.WORD	0		: NUMBER OF SEEKS PERFORMED
554	001314	000000		TIM.DN:	.WORD	0		: MINIMUM TIME
555	001316	000000			.WORD	0		: NUMBER OF COUNTS BELOW MIN. LIMIT
556	001320	000000			.WORD	0		: MAXIMUM TIME
557	001322	000000			.WORD	0		: NUMBER OF COUNTS ABOVE MAX. LIMIT
558	001324	000000	000000		.WORD	0,0		: TOTAL TIME OF ALL SEEKS
559	001330	000000			.WORD	0		: NUMBER OF SEEKS PERFORMED
560	001332	000000		TIM.PT:	.WORD	0		: POINTS TO TABLE OF TIMES
561	001334	000000		WCEFLG:	.WORD	0		: FATAL WRITE CHECK ERROR FLAG (TEST 20)
562	001336	000000		STALLO:	.WORD	0		: VARIABLE STALL (TEST 4)
563	001340	000000	000000	SVADR:	.WORD	0,0		: SAVE DISK ADDRESS (TEST 22)
564	001344	000000		SEKTR:	.WORD	0		: SEEK TIMER (TEST 10)
565	001346	000000		SEKCN:	.WORD	0		: SEEK COUNTER
566	001350	000000		DELTA:	.WORD	0		: TESTING RANGE FOR SERVO SETTLE DOWN TEST
567	001352	165000		TRCKWC:	.WORD	-(256.*22.)		: WORD COUNT FOR A FULL TRACK IN 16 BIT MODE
568	001354	000012		STALL1:	.WORD	10.		: 10 MILLISECOND STALL (TEST 0-11)
569	001356	000012		STALL2:	.WORD	10.		: 10 MILLISECOND STALL (TEST 16-21)
570	001360	011610		STALL3:	.WORD	5000.		: 5 SEC STALL (TEST 22)
571	001362	000031		MXSTAL:	.WORD	25.		: MAX. INCREMENTING STALL ALLOWED IN TEST 4
572	001364	144		ERR.CT:	.BYTE	100.		: NUMBER OF ERRORS ALLOWED IN TESTS 16 - 21 BEFORE GOING TO THE NEXT TEST
574	001365	000			.BYTE	0		: RESERVED
576				: ADDRESSES AND VECTORS				
577	001366	176700		RH.ADR:	.WORD	176700		: RH11-RH70 UNIBUS ADDRESS
578	001370	000254	000240	RHVEC:	.WORD	254,5*32.		: RH11-RH70 VECTOR ADDRESS AND PRIORITY
579	001374	000104	000106	PKV:	.WORD	104,106		: KW11-P VECTOR ADDRESS

COMMON TAGS

580 001400 172540
 581 001402 172542
 582 001404 172544
 583 001406 000100 000102
 584 001412 177546
 585 001414 177564
 586 001416 177566
 587 001420 177514
 588 001422 177516

PKCS: .WORD 172540
 PKB: .WORD 172542
 PKC: .WORD 172544
 LKV: .WORD 100,102
 LKS: .WORD 177546
 TPS: .WORD 177564
 TPB: .WORD 177566
 LPS: .WORD 177514
 LPB: .WORD 177516

:KW11-P CONTROL AND STATUS REG.
 :KW11-P COUNT SET BUFFER
 :KW11-P COUNTER
 :KW11-L VECTOR ADDRESS
 :KW11-L STATUS REGISTER
 :TTY PRINTER STATUS
 :TTY PRINTER BUFFER
 :LINE PRINTER STATUS
 :LINE PRINTER BUFFER

:BIT TABLE

591 001424 000001
 592 001426 000002
 593 001430 000004
 594 001432 000010
 595 001434 000020
 596 001436 000040
 597 001440 000100
 598 001442 000200
 599 001444 000400
 600 001446 001000
 601 001450 002000
 602 001452 004000
 603 001454 010000
 604 001456 020000
 605 001460 040000
 606 001462 100000
 607 001464 000001
 608 001466 000002
 609 001470 000004
 610 001472 000010
 611 001474 000020
 612 001476 000040
 613 001500 000100
 614 001502 000200

BITS: .WORD BIT00
 .WORD BIT01
 .WORD BIT02
 .WORD BIT03
 .WORD BIT04
 .WORD BIT05
 .WORD BIT06
 .WORD BIT07
 .WORD BIT08
 .WORD BIT09
 .WORD BIT10
 .WORD BIT11
 .WORD BIT12
 .WORD BIT13
 .WORD BIT14
 .WORD BIT15
 .WORD BIT00
 .WORD BIT01
 .WORD BIT02
 .WORD BIT03
 .WORD BIT04
 .WORD BIT05
 .WORD BIT06
 .WORD BIT07

:COMMON STORAGE FOR TEST PARAMETER

618 001504 000000
 619 001506 000000
 620 001510 000000
 621 001512 000000
 622 001514 000000
 623 001516 000000
 624 001520 000000
 625 001522 000000
 626 001524 000000
 627 001526 000000
 628 001530 000000
 629 001532 000000
 630 001534 000000

PRM: .WORD 0
 RPT: .WORD 0
 FC: .WORD 0
 LC: .WORD 0
 IC: .WORD 0
 FT: .WORD 0
 LT: .WORD 0
 IT: .WORD 0
 FS: .WORD 0
 LS: .WORD 0
 PAT: .WORD 0
 NC1: .WORD 0
 NC2: .WORD 0

:REPEAT COUNTS FOR ALL TESTS
 :FIRST CYLINDER
 :LAST CYLINDER
 :INCREMENT CYLINDER
 :FIRST TRACK
 :LAST TRACK
 :INCREMENT TRACK
 :FIRST SECTOR
 :LAST SECTOR
 :PATTERN CODE
 :NEW CYLINDER ADDRESS
 :NEW CYLINDER ADDRESS

:TABLE OF PARAMETER POINTERS

633 001536 002330
 634 001540 002344
 635 001542 002372

PRMPT: .WORD PRM0
 .WORD PRM1
 .WORD PRM2


```

692 001750 001177 000010 000000 .WORD 1177,10,0,256.,0,256.,1,0,0 ;STEPPING SEEK (T3)
693 001756 000400 000000 000400
694 001764 000001 000000 000000
695 001772 001177 000001 000000 .WORD 1177,1,0,410.,0,814.,1,0,0 ;OSCILLATING SEEK (T4)
696 002000 000632 000000 001456
697 002006 000001 000000 000000
698 002014 001177 000001 000000 .WORD 1177,1,0,410.,0,814.,1,0,0 ;CONVERGING/DIVERGING SEEK (T5)
699 002022 000632 000000 001456
700 002030 000001 000000 000000
701 002036 001177 000001 000000 .WORD 1177,1,0,410.,0,814.,1,0,0 ;SERVO ADDRESSING LOGIC NOISE (T6)
702 002044 000632 000000 001456
703 002052 000001 000000 000000
704 002060 000337 011610 000000 .WORD 337,5000.,0,410.,0,814.,0,18. ;RANDOM SEEK TEST (T7)
705 002066 000632 000000 001456
706 002074 000000 000022
707 002100 000177 000001 000000 .WORD 177,1,0,410.,0,814.,100.,0 ;SERVO SETTLE DOWN TEST (T10)
708 002106 000632 000000 001456
709 002114 000144 000000
710 002120 001177 000001 000000 .WORD 1177,1,0,410.,0,814.,1,0,0 ;ALL SEEKS TEST (T11)
711 002126 000632 000000 001456
712 002134 000001 000000 000000
713 002142 001113 000001 000000 .WORD 1113,1,0,0,0,0 ;ROTATIONAL SPEED TIMING TEST (T12)
714 002150 000000 000000 000000
715 002156 000037 000001 000000 .WORD 37,1,0,410.,0,814. ;ONE CYLINDER SEEK TIMING TEST (T13)
716 002164 000632 000000 001456
717 002172 000037 000001 002000 .WORD 37,1,0,136.,0,255. ;ACCESS TIME MEASUREMENT TEST (T14)
718 002200 000210 000000 000377
719 002206 000037 000001 000000 .WORD 37,1,0,410.,0,814. ;MAXIMUM SEEK TIMING TEST (T15)
720 002214 000632 000000 001456
721 002222 000113 000001 000000 .WORD 113,1,0,0,0 ;SECTOR ADDRESSING TEST (T16)
722 002230 000000 000000
723 002234 001013 000001 000000 .WORD 1013,1,0,0,0 ;TRACK ADDRESSING TEST (T17)
724 002242 000000 000000
725 002246 007777 000001 000000 .WORD 7777,1,0,410.,0,814.,64.,0,18.,1,1,0,177777 ;DATA TEST (T20)
726 002254 000632 000000 001456
727 002262 000100 000000 000022
728 002270 000001 000001 000000
729 002276 177777
730 002300 000037 047040 000000 .WORD 37,20000.,0,410.,0,814. ;EXERCISER (T21)
731 002306 000632 000000 001456
732 002314 000037 011610 000000 .WORD 37,5000.,0,136.,0,255. ;RPO4 ACCESS TIME ADJUSTMENT TEST (T22)
733 002322 000210 000000 000377

```

;PARAMETER TABLES

;RECAL/SEEK (T0)

```

PRM0: .WORD 1125
       .WORD 200.
       .WORD 410.
       .WORD 814.
       .WORD 0
       .WORD 0

```

;SEEK/SEEK (T1)

```

PRM1: .WORD 3377
       .WORD 100.

```

```

734
735
736
737
738 002330 001125
739 002332 000310
740 002334 000632
741 002336 001456
742 002340 000000
743 002342 000000
744
745
746 002344 003377
747 002346 000144

```


748 002350 000000
749 002352 000200
750 002354 000000
751 002356 000400
752 002360 000000
753 002362 000000
754 002364 000000
755 002366 000000
756 002370 000000
757
758
759 002372 001177
760 002374 000001
761 002376 000000
762 002400 000632
763 002402 000000
764 002404 001456
765 002406 000001
766 002410 000000
767 002412 000000
768
769
770 002414 001177
771 002416 000001
772 002420 000000
773 002422 000400
774 002424 000000
775 002426 001000
776 002430 000001
777 002432 000000
778 002434 000000
779
780
781 002436 001177
782 002440 000001
783 002442 000000
784 002444 000632
785 002446 000000
786 002450 001456
787 002452 000001
788 002454 000000
789 002456 000000
790
791
792 002460 001177
793 002462 000001
794 002464 000000
795 002466 000632
796 002470 000000
797 002472 001456
798 002474 000001
799 002476 000000
800 002500 000000
801
802
803 002502 001177

.WORD 0
.WORD 128.
.WORD 0
.WORD 256.
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0

: INCREMENT SEEK (T2)

PRM2: .WORD 1177
.WORD 1
.WORD 0
.WORD 410.
.WORD 0
.WORD 814.
.WORD 1
.WORD 0
.WORD 0

: STEPPING SEEK (T3)

PRM3: .WORD 1177
.WORD 1
.WORD 0
.WORD 256.
.WORD 0
.WORD 512.
.WORD 1
.WORD 0
.WORD 0

: OSCILLATING SEEK (T4)

PRM4: .WORD 1177
.WORD 1
.WORD 0
.WORD 410.
.WORD 0
.WORD 814.
.WORD 1
.WORD 0
.WORD 0

: CONVERGING/DIVERGING SEEK (T5)

PRM5: .WORD 1177
.WORD 1
.WORD 0
.WORD 410.
.WORD 0
.WORD 814.
.WORD 1
.WORD 0
.WORD 0

: SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)

PRM6: .WORD 1177

804 002504 000001
 805 002506 000000
 806 002510 000632
 807 002512 000000
 808 002514 001456
 809 002516 000001
 810 002520 000000
 811 002522 000000

.WORD 1
 .WORD 0
 .WORD 410.
 .WORD 0
 .WORD 814.
 .WORD 1
 .WORD 0
 .WORD 0

:RANDOM SEEK TEST (T7)

814 002524 000337
 815 002526 011610
 816 002530 000000
 817 002532 000632
 818 002534 000000
 819 002536 001456
 820 002540 000000
 821 002542 000022

PRM7: .WORD 337
 .WORD 5000.
 .WORD 0
 .WORD 410.
 .WORD 0
 .WORD 814.
 .WORD 0
 .WORD 18.

:SERVO SETTLE DOWN TEST (T10)

824 002544 000177
 825 002546 000001
 826 002550 000000
 827 002552 000632
 828 002554 000000
 829 002556 001456
 830 002560 000144
 831 002562 000000

PRM10: .WORD 177
 .WORD 1
 .WORD 0
 .WORD 410.
 .WORD 0
 .WORD 814.
 .WORD 100.
 .WORD 0

:ALL SEEKS TEST (T11)

834 002564 001177
 835 002566 000001
 836 002570 000000
 837 002572 000632
 838 002574 000000
 839 002576 001456
 840 002600 000001
 841 002602 000000
 842 002604 000000

PRM11: .WORD 1177
 .WORD 1
 .WORD 0
 .WORD 410.
 .WORD 0
 .WORD 814.
 .WORD 1
 .WORD 0
 .WORD 0

:ROTATIONAL SPEED TIMING TEST (T12)

845 002606 001113
 846 002610 000001
 847 002612 000000
 848 002614 000000
 849 002616 000000
 850 002620 000000

PRM12: .WORD 1113
 .WORD 1
 .WORD 0
 .WORD 0
 .WORD 0
 .WORD 0

:ONE CYLINDER SEEK TIMING TEST (T13)

853 002622 000037
 854 002624 000001
 855 002626 000000
 856 002630 000632
 857 002632 000000
 858 002634 001456
 859

PRM13: .WORD 37
 .WORD 1
 .WORD 0
 .WORD 410.
 .WORD 0
 .WORD 814.

860			: ACCESS TIME MEASUREMENT TEST (T14)
861	002636	000037	PRM14: .WORD 37
862	002640	000001	.WORD 1
863	002642	000000	.WORD 0
864	002644	000210	.WORD 136.
865	002646	000000	.WORD 0
866	002650	000377	.WORD 255.
867			
868			: MAXIMUM SEEK TIMING TEST (T15)
869	002652	000037	PRM15: .WORD 37
870	002654	000001	.WORD 1
871	002656	000000	.WORD 0
872	002660	000632	.WORD 410.
873	002662	000000	.WORD 0
874	002664	001456	.WORD 814.
875			
876			: SECTOR ADDRESSING TEST (T16)
877	002666	000113	PRM16: .WORD 113
878	002670	000001	.WORD 1
879	002672	000000	.WORD 0
880	002674	000000	.WORD 0
881	002676	000000	.WORD 0
882			
883			: TRACK ADDRESSING TEST (T17)
884	002700	001013	PRM17: .WORD 1013
885	002702	000001	.WORD 1
886	002704	000000	.WORD 0
887	002706	000000	.WORD 0
888	002710	000000	.WORD 0
889			
890			: DATA TEST (T20)
891	002712	007777	PRM20: .WORD 7777
892	002714	000001	.WORD 1
893	002716	000000	.WORD 0
894	002720	000632	.WORD 410.
895	002722	000000	.WORD 0
896	002724	001456	.WORD 814.
897	002726	000100	.WORD 64.
898	002730	000000	.WORD 0
899	002732	000632	.WORD 410.
900	002734	000001	.WORD 1
901	002736	000001	.WORD 1
902	002740	000000	.WORD 0
903	002742	177777	PTRN15: .WORD 177777
904			
905			: EXERCISER (T21)
906	002744	000037	PRM21: .WORD 37
907	002746	047040	.WORD 20000.
908	002750	000000	.WORD 0
909	002752	000632	.WORD 410.
910	002754	000000	.WORD 0
911	002756	001456	.WORD 814.
912			
913			: RPO4 ACCESS TIME ADJUSTMENT TEST (T22)
914	002760	000037	PRM22: .WORD 37
915	002762	011610	.WORD 5000.

916 002764 000000
917 002766 000210
918 002770 000000
919 002772 000377
920
921
922
923
924
925 002774 043370
926 002776 000000
927 003000 003142
928 003002 003244
929
930 003004 043370
931 003006 000000
932 003010 003131
933 003012 003255
934
935 003014 043422
936 003016 043471
937 003020 000000
938 003022 001750
939
940 003024 043506
941 003026 043554
942 003030 000000
943 003032 005670
944
945 003034 043571
946 003036 043633
947 003040 000000
948 003042 012120
949
950 003044 003104
951 003046 003144
952 003050 003204
953 003052 003244
954 003054 003304
955 003056 003344
956 003060 003404
957 003062 003444
958 003064 003504
959 003066 003544
960 003070 003604
961 003072 003644
962 003074 003704
963 003076 003744
964 003100 004004
965 003102 004044
966
967
968
969 003104 165555
970 003106 133333
971 003110 165555

;SEEK TIMING LIMITS

T7A: .WORD MSG7
.WORD 0
.WORD 1634. ;(16.67-2%)*#2
.WORD 1700. ;(16.67+2%)*#2
T7B: .WORD MSG7
.WORD 0
.WORD 1625. ;(16.67-2.5%)*#2
.WORD 1709. ;(16.67+2.5%)*#2
T10: .WORD MSG10A
.WORD MSG10B
.WORD 0 ;NO LOWER LIMIT
.WORD 1000. ;(7+3)*#2
T11: .WORD MSG11A
.WORD MSG11B
.WORD 0 ;NO LOWER LIMIT
.WORD 3000. ;(28+2)*#2
T12: .WORD MSG12A
.WORD MSG12B
.WORD 0 ;NO LOWER LIMIT
.WORD 5200. ;(50+2)*#2

PAT.PT: .WORD PAT0 ;TABLE OF POINTERS WHICH POINT TO THE
.WORD PAT1 ;PATTERNS USED BY THE DATA TEST
.WORD PAT2
.WORD PAT3
.WORD PAT4
.WORD PAT5
.WORD PAT6
.WORD PAT7
.WORD PAT8
.WORD PAT9
.WORD PAT10
.WORD PAT11
.WORD PAT12
.WORD PAT13
.WORD PAT14
.WORD PAT15

;PATTERNS 0 THRU 15

PAT0: .WORD 165555 ;PATTERN 0
.WORD 133333
.WORD 165555

972	003112	133333	.WORD	133333
973	003114	165555	.WORD	165555
974	003116	133333	.WORD	133333
975	003120	165555	.WORD	165555
976	003122	133333	.WORD	133333
977	003124	165555	.WORD	165555
978	003126	133333	.WORD	133333
979	003130	165555	.WORD	165555
980	003132	133333	.WORD	133333
981	003134	165555	.WORD	165555
982	003136	133333	.WORD	133333
983	003140	165555	.WORD	165555
984	003142	133333	.WORD	133333
985				
986	003144	000001	PAT1: .WORD	000001 ;PATTERN 1
987	003146	000003	.WORD	000003
988	003150	000007	.WORD	000007
989	003152	000017	.WORD	000017
990	003154	000037	.WORD	000037
991	003156	000077	.WORD	000077
992	003160	000177	.WORD	000177
993	003162	000377	.WORD	000377
994	003164	000777	.WORD	000777
995	003166	001777	.WORD	001777
996	003170	003777	.WORD	003777
997	003172	007777	.WORD	007777
998	003174	017777	.WORD	017777
999	003176	037777	.WORD	037777
1000	003200	077777	.WORD	077777
1001	003202	177777	.WORD	177777
1002				
1003	003204	177776	PAT2: .WORD	177776 ;PATTERN 2
1004	003206	177774	.WORD	177774
1005	003210	177770	.WORD	177770
1006	003212	177760	.WORD	177760
1007	003214	177740	.WORD	177740
1008	003216	177700	.WORD	177700
1009	003220	177600	.WORD	177600
1010	003222	177400	.WORD	177400
1011	003224	177000	.WORD	177000
1012	003226	176000	.WORD	176000
1013	003230	174000	.WORD	174000
1014	003232	170000	.WORD	170000
1015	003234	160000	.WORD	160000
1016	003236	140000	.WORD	140000
1017	003240	100000	.WORD	100000
1018	003242	000000	.WORD	000000
1019				
1020	003244	000000	PAT3: .WORD	000000 ;PATTERN 3
1021	003246	000000	.WORD	000000
1022	003250	000000	.WORD	000000
1023	003252	177777	.WORD	177777
1024	003254	177777	.WORD	177777
1025	003256	177777	.WORD	177777
1026	003260	000000	.WORD	000000
1027	003262	000000	.WORD	000000

1028	003264	177777	.WORD	177777
1029	003266	177777	.WORD	177777
1030	003270	000000	.WORD	000000
1031	003272	177777	.WORD	177777
1032	003274	000000	.WORD	000000
1033	003276	177777	.WORD	177777
1034	003300	000000	.WORD	000000
1035	003302	177777	.WORD	177777

PAT4: .WORD 000000 ;PATTERN 4

1036				
1037	003304	000000	.WORD	000000
1038	003306	010421	.WORD	010421
1039	003310	021042	.WORD	021042
1040	003312	031463	.WORD	031463
1041	003314	042104	.WORD	042104
1042	003316	052525	.WORD	052525
1043	003320	063146	.WORD	063146
1044	003322	073567	.WORD	073567
1045	003324	104210	.WORD	104210
1046	003326	114631	.WORD	114631
1047	003330	125252	.WORD	125252
1048	003332	135673	.WORD	135673
1049	003334	146314	.WORD	146314
1050	003336	156735	.WORD	156735
1051	003340	167356	.WORD	167356
1052	003342	177777	.WORD	177777

PAT5: .WORD 052525 ;PATTERN 5

1053				
1054	003344	052525	.WORD	052525
1055	003346	052525	.WORD	052525
1056	003350	052525	.WORD	052525
1057	003352	125252	.WORD	125252
1058	003354	125252	.WORD	125252
1059	003356	125252	.WORD	125252
1060	003360	052525	.WORD	052525
1061	003362	052525	.WORD	052525
1062	003364	125252	.WORD	125252
1063	003366	125252	.WORD	125252
1064	003370	052525	.WORD	052525
1065	003372	125252	.WORD	125252
1066	003374	052525	.WORD	052525
1067	003376	125252	.WORD	125252
1068	003400	052525	.WORD	052525
1069	003402	125252	.WORD	125252

PAT6: .WORD 007417 ;PATTERN 6

1070				
1071	003404	007417	.WORD	007417
1072	003406	007417	.WORD	007417
1073	003410	007417	.WORD	007417
1074	003412	170360	.WORD	170360
1075	003414	170360	.WORD	170360
1076	003416	170360	.WORD	170360
1077	003420	007417	.WORD	007417
1078	003422	007417	.WORD	007417
1079	003424	170360	.WORD	170360
1080	003426	170360	.WORD	170360
1081	003430	007417	.WORD	007417
1082	003432	170360	.WORD	170360
1083	003434	007417	.WORD	007417

K02

1084	003436	170360	.WORD	170360	
1085	003440	007417	.WORD	007417	
1086	003442	170360	.WORD	170360	
1087					
1088	003444	026455	PAT7: .WORD	026455	;PATTERN 7
1089	003446	026455	.WORD	026455	
1090	003450	026455	.WORD	026455	
1091	003452	151322	.WORD	151322	
1092	003454	151322	.WORD	151322	
1093	003456	151322	.WORD	151322	
1094	003460	026455	.WORD	026455	
1095	003462	026455	.WORD	026455	
1096	003464	151322	.WORD	151322	
1097	003466	151322	.WORD	151322	
1098	003470	026455	.WORD	026455	
1099	003472	151322	.WORD	151322	
1100	003474	026455	.WORD	026455	
1101	003476	151322	.WORD	151322	
1102	003500	026455	.WORD	026455	
1103	003502	151322	.WORD	151322	
1104					
1105	003504	165555	PAT8: .WORD	165555	;PATTERN 8
1106	003506	133333	.WORD	133333	
1107	003510	165555	.WORD	165555	
1108	003512	133333	.WORD	133333	
1109	003514	165555	.WORD	165555	
1110	003516	133333	.WORD	133333	
1111	003520	165555	.WORD	165555	
1112	003522	133333	.WORD	133333	
1113	003524	165555	.WORD	165555	
1114	003526	133333	.WORD	133333	
1115	003530	165555	.WORD	165555	
1116	003532	133333	.WORD	133333	
1117	003534	165555	.WORD	165555	
1118	003536	133333	.WORD	133333	
1119	003540	165555	.WORD	165555	
1120	003542	133333	.WORD	133333	
1121					
1122	003544	000001	PAT9: .WORD	000001	;PATTERN 9
1123	003546	000002	.WORD	000002	
1124	003550	000004	.WORD	000004	
1125	003552	000010	.WORD	000010	
1126	003554	000020	.WORD	000020	
1127	003556	000040	.WORD	000040	
1128	003560	000100	.WORD	000100	
1129	003562	000200	.WORD	000200	
1130	003564	000400	.WORD	000400	
1131	003566	001000	.WORD	001000	
1132	003570	002000	.WORD	002000	
1133	003572	004000	.WORD	004000	
1134	003574	010000	.WORD	010000	
1135	003576	020000	.WORD	020000	
1136	003600	040000	.WORD	040000	
1137	003602	100000	.WORD	100000	
1138					
1139	003604	177776	PAT10: .WORD	177776	;PATTERN 10

1140	003606	177775	.WORD	177775
1141	003610	177773	.WORD	177773
1142	003612	177767	.WORD	177767
1143	003614	177757	.WORD	177757
1144	003616	177737	.WORD	177737
1145	003620	177677	.WORD	177677
1146	003622	177577	.WORD	177577
1147	003624	177377	.WORD	177377
1148	003626	176777	.WORD	176777
1149	003630	175777	.WORD	175777
1150	003632	173777	.WORD	173777
1151	003634	167777	.WORD	167777
1152	003636	157777	.WORD	157777
1153	003640	137777	.WORD	137777
1154	003642	077777	.WORD	077777
1155				
1156	003644	172666	PAT11: .WORD	172666
1157	003646	155555	.WORD	155555
1158	003650	172666	.WORD	172666
1159	003652	155555	.WORD	155555
1160	003654	172666	.WORD	172666
1161	003656	155555	.WORD	155555
1162	003660	172666	.WORD	172666
1163	003662	155555	.WORD	155555
1164	003664	172666	.WORD	172666
1165	003666	155555	.WORD	155555
1166	003670	172666	.WORD	172666
1167	003672	155555	.WORD	155555
1168	003674	172666	.WORD	172666
1169	003676	155555	.WORD	155555
1170	003700	172666	.WORD	172666
1171	003702	155555	.WORD	155555
1172				
1173	003704	077777	PAT12: .WORD	077777
1174	003706	137777	.WORD	137777
1175	003710	157777	.WORD	157777
1176	003712	167777	.WORD	167777
1177	003714	173777	.WORD	173777
1178	003716	175777	.WORD	175777
1179	003720	176777	.WORD	176777
1180	003722	177377	.WORD	177377
1181	003724	177577	.WORD	177577
1182	003726	177677	.WORD	177677
1183	003730	177737	.WORD	177737
1184	003732	177757	.WORD	177757
1185	003734	177767	.WORD	177767
1186	003736	177773	.WORD	177773
1187	003740	177775	.WORD	177775
1188	003742	177776	.WORD	177776
1189				
1190	003744	153333	PAT13: .WORD	153333
1191	003746	066667	.WORD	066667
1192	003750	153333	.WORD	153333
1193	003752	066667	.WORD	066667
1194	003754	153333	.WORD	153333
1195	003756	066667	.WORD	066667

;PATTERN 11

;PATTERN 12

;PATTERN 13

1196	003760	153333	.WORD	153333
1197	003762	066667	.WORD	066667
1198	003764	153333	.WORD	153333
1199	003766	066667	.WORD	066667
1200	003770	153333	.WORD	153333
1201	003772	066667	.WORD	066667
1202	003774	153333	.WORD	153333
1203	003776	066667	.WORD	066667
1204	004000	153333	.WORD	153333
1205	004002	066667	.WORD	066667

1206				
1207	004004	000000	PAT14: .WORD	000000 ;PATTERN 14
1208	004006	177777	.WORD	177777
1209	004010	177777	.WORD	177777
1210	004012	177777	.WORD	177777
1211	004014	177777	.WORD	177777
1212	004016	177777	.WORD	177777
1213	004020	177777	.WORD	177777
1214	004022	177777	.WORD	177777
1215	004024	177777	.WORD	177777
1216	004026	177777	.WORD	177777
1217	004030	177777	.WORD	177777
1218	004032	177777	.WORD	177777
1219	004034	177777	.WORD	177777
1220	004036	177777	.WORD	177777
1221	004040	177777	.WORD	177777
1222	004042	177777	.WORD	177777

1223				
1224	004044	177777	PAT15: .WORD	177777 ;PATTERN 15
1225	004046	000000	.WORD	000000
1226	004050	000000	.WORD	000000
1227	004052	000000	.WORD	000000
1228	004054	000000	.WORD	000000
1229	004056	000000	.WORD	000000
1230	004060	000000	.WORD	000000
1231	004062	000000	.WORD	000000
1232	004064	000000	.WORD	000000
1233	004066	000000	.WORD	000000
1234	004070	000000	.WORD	000000
1235	004072	000000	.WORD	000000
1236	004074	000000	.WORD	000000
1237	004076	000000	.WORD	000000
1238	004100	000000	.WORD	000000
1239	004102	000000	.WORD	000000

1240
1241 ;DPB (DATA PARAMETER BLOCK)

1242				
1243	004104	000	DPB.A: .BYTE	0
1244	004105	000	.BYTE	0
1245	004106	000	.BYTE	0
1246	004107	000	.BYTE	0
1247	004110	000000	.WORD	0
1248	004112	047662	.WORD	BUFFER
1249				
1250	004114	000	.BYTE	0
1251				

:: (0) DRIVE NUMBER
 :: (1) OFFSET VALUE OR FMT22, ECI, AND HCI
 :: (2) COMMAND
 :: (3) PSEL AND A17 AND A16
 :: (4) WORD COUNT (MUST BE NEG.)
 :: (6) BUFFER ADDRESS OR REGISTER TABLE POINTER
 :: (10) SECTOR ADDRESS OR FIRST REG. INDEX

Address	Field 1	Field 2	Field 3	Field 4	Field 5	Description
1252	004115	000	.BYTE	0		(11) TRACK ADDRESS OR LAST REG. INDEX
1253						
1254	004116	000000	.WORD	0		(12) CYLINDER ADDRESS
1255	004120	004204	.WORD	RP.REG		(14) ERROR TABLE POINTER
1256						POINTS TO THE FIRST OF TWENTY LOCATIONS OF WHERE THE DRIVER IS TO STORE THE RH11/RP04 REGISTERS ON AN ERROR. IF LEFT ZERO REGISTERS ARE NOT SAVED.
1257						
1258						
1259						
1260						
1261	004122	000000	.WORD	0		(16) STATUS/ERROR INDICATOR
1262						BIT15=1=>ERROR OCCURRED
1263						BIT07=1=>DONE
1264						BIT14-BIT09 AND BIT06-BIT03
1265						INDICATE TYPE OF ERROR
1266						
1267	004124	000	DPB.B: .BYTE	0		(0) DRIVE NUMBER
1268	004125	000	.BYTE	0		(1) OFFSET VALUE OR FMT22, ECI, AND HCI
1269	004126	000	.BYTE	0		(2) COMMAND
1270	004127	000	.BYTE	0		(3) PSEL AND A17 AND A16
1271	004130	177776	.WORD	-2		(4) WORD COUNT (MUST BE NEG.)
1272	004132	047662	.WORD	BUFFER		(6) BUFFER ADDRESS OR REGISTER TABLE POINTER
1273						
1274	004134	000	.BYTE	0		(10) SECTOR ADDRESS OR FIRST REG. INDEX
1275						
1276	004135	000	.BYTE	0		(11) TRACK ADDRESS OR LAST REG. INDEX
1277						
1278	004136	000000	.WORD	0		(12) CYLINDER ADDRESS
1279	004140	004204	.WORD	RP.REG		(14) ERROR TABLE POINTER
1280						POINTS TO THE FIRST OF TWENTY LOCATIONS OF WHERE THE DRIVER IS TO STORE THE RH11/RP04 REGISTERS ON AN ERROR. IF LEFT ZERO REGISTERS ARE NOT SAVED.
1281						
1282						
1283						
1284						
1285	004142	000000	.WORD	0		(16) STATUS/ERROR INDICATOR
1286						BIT15=1=>ERROR OCCURRED
1287						BIT07=1=>DONE
1288						BIT14-BIT09 AND BIT06-BIT03
1289						INDICATE TYPE OF ERROR
1290						
1291	004144	000	DPB.C: .BYTE	0		(0) DRIVE NUMBER
1292	004145	000	.BYTE	0		(1) OFFSET VALUE OR FMT22, ECI, AND HCI
1293	004146	000	.BYTE	0		(2) COMMAND
1294	004147	000	.BYTE	0		(3) PSEL AND A17 AND A16
1295	004150	177776	.WORD	-2		(4) WORD COUNT (MUST BE NEG.)
1296	004152	047662	.WORD	BUFFER		(6) BUFFER ADDRESS OR REGISTER TABLE POINTER
1297						
1298	004154	000	.BYTE	0		(10) SECTOR ADDRESS OR FIRST REG. INDEX
1299						
1300	004155	000	.BYTE	0		(11) TRACK ADDRESS OR LAST REG. INDEX
1301						
1302	004156	000000	.WORD	0		(12) CYLINDER ADDRESS
1303	004160	004204	.WORD	RP.REG		(14) ERROR TABLE POINTER
1304						POINTS TO THE FIRST OF TWENTY LOCATIONS OF WHERE THE DRIVER IS TO STORE THE RH11/RP04 REGISTERS ON AN ERROR. IF LEFT
1305						
1306						
1307						

1308					:ZERO REGISTERS ARE NOT SAVED.	
1309	004162	000000	.WORD	0	:(16) STATUS/ERROR INDICATOR	
1310					:BIT15=1=>ERROR OCCURRED	
1311					:BIT07=1=>DONE	
1312					:BIT14-BIT09 AND BIT06-BIT03	
1313					:INDICATE TYPE OF ERROR	
1314						
1315	004164	000	DTADPB:	.BYTE	0	:(0) DRIVE NUMBER
1316	004165	000		.BYTE	0	:(1) OFFSET VALUE OR FMT22, ECT, AND HCI
1317	004166	000		.BYTE	0	:(2) COMMAND
1318	004167	000		.BYTE	0	:(3) PSEL AND A17 AND A16
1319	004170	000000		.WORD	0	:(4) WORD COUNT (MUST BE NEG.)
1320	004172	047662		.WORD	BUFFER	:(6) BUFFER ADDRESS OR
1321						REGISTER TABLE POINTER
1322	004174	000		.BYTE	0	:(10) SECTOR ADDRESS OR
1323						FIRST REG. INDEX
1324	004175	000		.BYTE	0	:(11) TRACK ADDRESS OR
1325						LAST REG. INDEX
1326	004176	000000		.WORD	0	:(12) CYLINDER ADDRESS
1327	004200	004204		.WORD	RP.REG	:(14) ERROR TABLE POINTER
1328						POINTS TO THE FIRST OF TWENTY
1329						LOCATIONS OF WHERE THE DRIVER
1330						IS TO STORE THE RH11/RP04
1331						REGISTERS ON AN ERROR. IF LEFT
1332						ZERO REGISTERS ARE NOT SAVED.
1333	004202	000000		.WORD	0	:(16) STATUS/ERROR INDICATOR
1334						:BIT15=1=>ERROR OCCURRED
1335						:BIT07=1=>DONE
1336						:BIT14-BIT09 AND BIT06-BIT03
1337						:INDICATE TYPE OF ERROR

;SAVE RH11/RP04 REGISTERS HERE ON ERROR

1343	004204	000000	RP.REG:	.WORD	0	:RPCS1 (776700) CONTROL & STATUS #1
1344	004206	000000		.WORD	0	:RPMC (776702) WORD COUNT
1345	004210	000000		.WORD	0	:RPBA (776704) BUS ADDRESS
1346	004212	000000		.WORD	0	:RPDA (776706) DESIRED SECTOR/TRACK
1347	004214	000000		.WORD	0	:RPCS2 (776710) CONTROL & STATUS #2
1348	004216	000000		.WORD	0	:RPDS1 (776712) DISK STATUS
1349	004220	000000		.WORD	0	:RPER1 (776714) ERROR REG. #1
1350	004222	000000		.WORD	0	:RPAS (776716) ATTENTION SUMMARY
1351	004224	000000		.WORD	0	:RPLA (776720) LOOK AHEAD
1352	004226	000000		.WORD	0	:RPDB (776722) DATA BUFFER
1353	004230	000000		.WORD	0	:RPMR (776724) MAINTAINABILITY
1354	004232	000000		.WORD	0	:RPDT (776726) DRIVE TYPE
1355	004234	000000		.WORD	0	:RPSN (776730) SERIAL NUMBER
1356	004236	000000		.WORD	0	:RPOF (776732) OFFSET
1357	004240	000000		.WORD	0	:RPCA (776734) DESIRED CYLINDER
1358	004242	000000		.WORD	0	:RPCC (776736) CURRENT CYLINDER
1359	004244	000000		.WORD	0	:RPER2 (776740) ERROR REG #2
1360	004246	000000		.WORD	0	:RPER3 (776742) ERROR REG #3
1361	004250	000000		.WORD	0	:RPEC1 (776744) ECC POSITION
1362	004252	000000		.WORD	0	:RPEC2 (776746) ECC PATTERN

1364		
1365	004254	044672
1366	004256	044734
1367	004260	044765
1368	004262	045007
1369	004264	045035
1370	004266	045060
1371	004270	045117
1372	004272	045161
1373	004274	045225
1374	004276	045275
1375	004300	045315
1376	004302	045365
1377	004304	045435

:STATUS/ERROR MESSAGE POINTER TABLE

STATBL:	.WORD	MSG814	:OFFLINE OR UNSAFE DRIVE REQUESTED
	.WORD	MSG813	:UNLOAD DRIVE REQUESTED
	.WORD	MSG812	:PERSISTENT UNSAFE
	.WORD	MSG811	:PARITY ERROR OCCURRED
	.WORD	MSG810	:FATAL PARITY ERROR
	.WORD	MSG809	:SOFTWARE TIMEOUT ON THIS DRIVE
	.WORD	MSG808	:SOFTWARE TIMEOUT ON ANOTHER DRIVE
	.WORD	MSG806	:ERROR OCCURRED DURING I/O OPERATION
	.WORD	MSG805	:ERROR OCCURRED DURING NON-I/O OPERATION
	.WORD	MSG804	:UNSAFE OCCURRED
	.WORD	MSG803	:AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
	.WORD	MSG802	:DRIVE HAS NOT RESPONDED TO PORT REQUEST
	.WORD	MSG801	:DRIVE HAS BECOME NONEXISTENT

ERROR POINTER TABLE

.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

1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433

004306

SERRTB:
 *EM AND DH ARE ASCIZ MESSAGES, DT IS A STRING OF WORDS THAT POINT TO THE
 *DATA TO BE TYPED AND DF IS A STRING OF DATA THAT TELL HOW THE DT WORDS
 *ARE TO BE TYPED. IF ANY OF THE POINTERS ARE NOT NEEDED FOR A PARTICULAR
 *ERROR IT IS REPLACED WITH A ZERO.
 *EACH OF THE ITEMS BELOW REFER TO THE ERROR NUMBER AND INDICATE
 *THE INFORMATION THAT WILL BE TYPED WHEN THE ERROR OCCURS.
 *UNLESS STATED OTHER ALL NUMBERS ARE OCTAL

* ERROR ITEM 1
 * RH11 INTERRUPT OCCURED (RPAS = 0)
 * ERR PC RPAS
 * SERRPC SREG3

004306 044013
 004310 045473
 004312 047054
 004314 047506

EM1
DH1
DT1
DF1

* ERROR ITEM 2
 * UNEXPECTED ATTENTION OCCURRED
 * ERR PC DRIVE RPAS RPOS1 RPER1 RPER2 RPER3
 * SERRPC SREG1 SREG3 RPERRS RPERRS+2 RPERRS+4 RPERRS+6

004316 044056
 004320 045510
 004322 047060
 004324 047512

EM2
DH2
DT2
DF2

* ERROR ITEM 3
 * MASSBUS PARITY ERROR (MCPE=1)
 * TEST ERR PC ADDRESS DATA
 * STMPO SERRPC RD.ADR RD.WRD

004326 044114
 004330 045576
 004332 047076
 004334 047516

EM3
DH3
DT3
DF3

* ERROR ITEM 4
 * MASSBUS PARITY ERROR (PAR=1)
 * TEST ERR PC ADDRESS GDATA BDDATA

E03

MD-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 31
DZRJA.014 19-APR-76 00:00 ERROR POINTER TABLE

Address	Code	Value	Description
1434			;* STMPD SERRPC WRT.ADR WRT.WD RD.WRD
1435			EM4
1436	004336	044151	DH4
1437	004340	045633	DT4
1438	004342	047106	DF4
1439	004344	047522	
1440			*** ERROR ITEM 5
1441			*** ADDRESS PLUG CHANGE BIT SET
1442			*** ERR PC DRIVE RPAS RPDS1 RPER1 RPER2 RPER3
1443			*** SERRPC SREG1 SREG3 RPERRS RPERRS+2 RPERRS+4 RPERRS+6
1444			EM5
1445	004346	044205	DH2
1446	004350	045510	DT2
1447	004352	047060	DF2
1448	004354	047512	
1449			;* ERROR ITEM 6 -- NOT USED
1450			0
1451	004356	000000	0
1452	004360	000000	0
1453	004362	000000	0
1454	004364	000000	0
1455			;* ERROR ITEM 7 -- NOT USED
1456			0
1457			0
1458			0
1459			0
1460	004366	000000	0
1461	004370	000000	0
1462	004372	000000	0
1463	004374	000000	0
1464			*** ERROR ITEM 10
1465			*** RH11/RPO4/5/6 FAILED TO RESPOND TO ADDRESSING
1466			*** RPCS1 ERR PC
1467			*** RH.ADR SERRPC
1468			EM10
1469			DH10
1470	004376	044241	DT10
1471	004400	045702	DF10
1472	004402	047140	
1473	004404	047526	
1474			*** ERROR ITEM 11
1475			*** DRIVE SELECTED IS NOT ONLINE
1476			*** DRIVE ERR PC
1477			*** SREG2 SERRPC
1478			EM11
1479			DH11
1480	004406	044317	DT11
1481	004410	045721	DF11
1482	004412	047144	
1483	004414	047532	
1484			*** ERROR ITEM 12
1485			*** IMPROPER HEADER DATA
1486			*** TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR
1487			*** STMPD SERRPC SREG0 CHKDRV CYL.DS TRK.DS SEC.DS
1488			*** GDCYL GDTRK GDCTR BDCYL BOTRK BDSCTR
1489			

F03

Line No.	Code	Address	Description
1490			::* CYL.DS TRK.DS SEC.DS CYL.RD TRK.RD SEC.RD
1491			::* CYLNR, TRACK, AND SECTOR ARE DECIMAL
1492	004416	044354	EM12
1493	004420	045740	DH12
1494	004422	047150	DT12
1495	004424	047536	DF12
1498			::* ERROR ITEM 13
1499			::* DATA COMPARE FAILURE
1500			::* TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR
1501			::* STMPO SERRPC SREG0 CHKDRV CYL.DS TRK.DS SEC.DS
1502			::* GDDAT BDDAT WRDCNT GDADR BDADR
1503			::* SGDDAT SBDAT SREG4 SGDADR SBDADR
1504			::* CYLNR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
1505			
1506	004426	044401	EM13
1507	004430	045740	DH12
1508	004432	047202	DT13
1509	004434	047546	DF13
1510			
1511			::* ERROR ITEM 14 -- FOLLOWS #13
1512			::* SGDDAT SBDAT SREG4 SGDADR SBDADR
1513			
1514	004436	000000	0
1515	004440	000000	0
1516	004442	047220	DT13A
1517	004444	047556	DF14
1518			
1519			::* ERROR ITEM 15
1520			::* DATA COMPARE FAILURE
1521			::* TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR
1522			::* STMPO SERRPC SREG0 CHKDRV CYL.DS TRK.DS SEC.DS
1523			::* GDDAT BDDAT WRDCNT GDADR BDADR
1524			::* SGDDAT SBDAT SREG4 SGDADR SBDADR
1525			::* CYLNR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
1526			
1527	004446	044401	EM13
1528	004450	045740	DH12
1529	004452	047202	DT13
1530	004454	047546	DF13
1531			
1532			::* ERROR ITEM 16 -- FOLLOWS #15
1533			::* SGDDAT SBDAT SREG4 SGDADR SBDADR
1534			
1535	004456	000000	0
1536	004460	000000	0
1537	004462	047220	DT13A
1538	004464	047556	DF14
1539			
1540			::* ERROR ITEM 17
1541			::* DISK ERROR IN TIMING TEST
1542			::* TEST ERR PC DRIVE RPS1 RPS1 RPER1 RPER2 RPER3
1543			::* STMPO SERRPC CHKDRV RP.REG RP.REG+12 RP.REG+14 RP.REG+40 RP.REG+42
1544			
1545	004466	044426	EM17

1546	004470	046154	DH17
1547	004472	047232	DT17
1548	004474	047562	DF17
1549			
1550			::* ERROR ITEM 20
1551			::* CLOCK (KW11-P) OVERFLOW IN TIMING TEST
1552			::* TEST ERR PC DRIVE RPCS1 RPS1 RPER1 RPER2 RPER3
1553			::* STMPO SERRPC CHKDRV RP.REG RP.REG+12 RP.REG+14 RP.REG+40 RP.REG+42
1554			
1555	004476	044460	EM20
1556	004500	046154	DH17
1557	004502	047232	DT17
1558	004504	047562	DF17
1559			
1560			::* ERROR ITEM 21
1561			::* DATA COMPARE FAILURE
1562			::* TEST ERR PC TST PC DRIVE CYLNDR TRACK
1563			::* STMPO SERRPC SREG0 CHKDRV CYL.DS TRK.DS
1564			::* GODAT BDDAT WRDCNT SECTOR
1565			::* SREG1 SBDDAT SREG4 SREG1
1566			::* CYLINDER, TRACK, WRDCNT, AND SECTOR ARE DECIMAL
1567			
1568	004506	044401	EM13
1569	004510	046252	DH21
1570	004512	047252	DT21
1571	004514	047566	DF21
1572			
1573			::* ERROR ITEM 22--FOLLOWS #21
1574			::* SREG1 SBDDAT SREG4 SREG1
1575			
1576	004516	000000	0
1577	004520	000000	0
1578	004522	047266	DT21A
1579	004524	047576	DF22
1580			
1581			::* ERROR ITEM 23
1582			::* DISK ERROR DURING SEEK
1583			::* TEST ERR PC DRIVE CYLNDR RPCS1 RPCS2 RPS1
1584			::* STMPO SERRPC CHKDRV CYL.DS RP.REG RP.REG+10 RP.REG+12
1585			::* RPER1 RPER2 RPER3 RPCA RPCC
1586			::* RP.REG+14 RP.REG+40 RP.REG+42 RP.REG+34 RP.REG+36
1587			
1588	004526	044527	EM23
1589	004530	046367	DH23
1590	004532	047276	DT23
1591	004534	047602	DF23
1592			
1593			::* ERROR ITEM 24
1594			::* SEEK NOT COMPLETE WITHIN 120 MS
1595			::* TEST ERR PC DRIVE CYLNDR RPCS1 RPCS2 RPS1
1596			::* STMPO SERRPC CHKDRV CYL.DS RP.REG RP.REG+10 RP.REG+12
1597			::* RPER1 RPER2 RPER3 RPCA RPCC
1598			::* RP.REG+14 RP.REG+40 RP.REG+42 RP.REG+34 RP.REG+36
1599			
1600	004536	044556	EM24
1601	004540	046367	DH23

1602 004542 047276
1603 004544 047602

DT23
DF23

1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625

```
*****  
*****  
* ERROR ITEMS 23-40 NOT USED  
* ERROR ITEMS 41-46 WILL HAVE AN EM THAT  
* VARIES DEPENDING ON THE ERROR, IT WILL BE IN THE FORM:  
* RH11/RPO4/5/6 ERROR (MESSAGE)  
* WHERE MESSAGE WILL BE ONE OR MORE OF THE FOLLOWING:  
* 1) OFFLINE OR UNSAFE DRIVE REQUESTED  
* 2) UNLOADED DRIVE REQUESTED  
* 3) PERSISTENT UNSAFE  
* 4) PARITY ERROR OCCURRED  
* 5) FATAL PARITY ERROR  
* 6) SOFTWARE TIMEOUT ON THIS DRIVE  
* 7) SOFTWARE TIMEOUT ON ANOTHER DRIVE  
* 8) ERROR OCCURRED DURING I/O OPERATION  
* 9) ERROR OCCURRED DURING NON-I/O OPERATION  
* 10) UNSAFE OCCURRED  
* 11) AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
```

1626 004546

```
ITEM41:  
* ERROR ITEM 41  
* RH11/RPO4/5/6 ERROR (MESSAGE)  
* TEST ERR PC TST PC DRIVE  
* STMPO SERRPC SREGO CHKDRV
```

1627
1628
1629
1630
1631
1632
1633 004546 044616
1634 004550 046522
1635 004552 047326
1636 004554 047612

EM41
DM41
DT41
DF41

1637
1638
1639
1640
1641
1642

```
* ERROR ITEM 42  
* RH11/RPO4/5/6 ERROR (MESSAGE)  
* TEST ERR PC TST PC DRIVE RPCS1 RPCS2 RPDS1  
* STMPO SERRPC SREGO CHKDRV RP.REG RP.REG+10 RP.REG+12
```

1643 004556 044616
1644 004560 046560
1645 004562 047336
1646 004564 047616

EM41
DM42
DT42
DF42

1647
1648
1649
1650
1651
1652
1653

```
* ERROR ITEM 43  
* RH11/RPO4/5/6 ERROR (MESSAGE)  
* TEST ERR PC TST PC DRIVE RPCS1 RPCS2 RPDS1  
* STMPO SERRPC SREGO CHKDRV RP.REG RP.REG+10 RP.REG+12  
* RPER1 RPER2 RPER3  
* RP.REG+14 RP.REG+40 RP.REG+42
```

1654
1655 004566 044616
1656 004570 046560
1657 004572 047354

EM41
DM42
DT43

1658 004574 047622

DF43

```

    * ERROR ITEM 44
    * RH11/RPO4/5/6 ERROR (MESSAGE)
    * TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
    * STMPO SERRPC SREGO CHKDRV CYL.DS TRK.DS SEC.DS
    * RPCS1 RPCS2 RPDS1 RPCC RPCA RPOA
    * RP.REG RP.REG+10 RP.REG+12 RP.REG+36 RP.REG+34 RP.REG+06
    * RPER1 RPER2 RPER3
    * RP.REG+14 RP.REG+40 RP.REG+42
    * CYLNDR, TRACK, AND SECTOR ARE DECIMAL
    
```

1670 004576 044616

EM41

1671 004600 045740

DH12

1672 004602 047400

DT44

1673 004604 047632

DF44

1674

```

    * ERROR ITEM 45
    * RH11/RPO4/5/6 ERROR (MESSAGE)
    * TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
    * STMPO SERRPC SREGO CHKDRV CYL.DS TRK.DS SEC.DS
    * RPCS1 RPCS2 RPDS1 RPCC RPCA RPOA
    * RP.REG RP.REG+10 RP.REG+12 RP.REG+36 RP.REG+34 RP.REG+06
    * RPER1 RPER2 RPER3 RPMC RPBA RPDB
    * RP.REG+14 RP.REG+40 RP.REG+42 RP.REG+2 RP.REG+4 RP.REG+22
    * CYLNDR, TRACK, AND SECTOR ARE DECIMAL
    
```

1685 004606 044616

EM41

1686 004610 045740

DH12

1687 004612 047440

DT45

1688 004614 047646

DF45

1689

```

    * ERROR ITEM 46
    * FATAL WRITE CHECK ERROR (MESSAGE)
    * TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
    * STMPO SERRPC SREGO CHKDRV CYL.DS TRK.DS SEC.DS
    * RPCS1 RPCS2 RPDS1 RPCC RPCA RPOA
    * RP.REG RP.REG+10 RP.REG+12 RP.REG+36 RP.REG+34 RP.REG+06
    * RPER1 RPER2 RPER3 RPMC RPBA RPDB
    * RP.REG+14 RP.REG+40 RP.REG+42 RP.REG+2 RP.REG+4 RP.REG+22
    * CYLNDR, TRACK, AND SECTOR ARE DECIMAL
    
```

1700 004616 044642

EM46

1701 004620 045740

DH12

1702 004622 047440

DT45

1703 004624 047646

DF45

1704


```

1705
1706          .SBTTL  START OF PROGRAM
1707
1708
1709 004626 012737 177777 001226 START3: MOV    #-1, @#BUSADR ;GET BUSADR FLAG
1710 004634 000402          BR    STRT1A
1711 004636 005037 001226 START1: CLR    @#BUSADR ;CLR BUSADR FLAG
1712 004642 005037 001224 STRT1A: CLR    @#CNTRLC ;NO CONTROL "C"
1713 004646 000411          BR    START
1714 004650 012737 177777 001226 START4: MOV    #-1, @#BUSADR ;SET BUSADR FLAG
1715 004656 000402          BR    STRT2A
1716 004660 005037 001226 START2: CLR    @#BUSADR ;CLR BUSADR FLAG
1717 004664 012737 177777 001224 STRT2A: MOV    #-1, @#CNTRLC ;SET CONTROL "C" FLAG
1718 004672 000005          START: RESET
1719          .SBTTL  INITIALIZE THE COMMON TAGS
1720          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1721 004674 012706 001100          MOV    $CMTAG, R6 ;;FIRST LOCATION TO BE CLEARED
1722 004700 005026          CLR    (R6)+ ;;CLEAR MEMORY LOCATION
1723 004702 022706 001140          CMP    #SWR, R6 ;;DONE?
1724 004706 001374          BNE    -6 ;;LOOP BACK IF NO
1725 004710 012706 001100          MOV    #STACK, SP ;;SETUP THE STACK POINTER
1726          ;;INITIALIZE A FEW VECTORS
1727 004714 012737 022564 000020          MOV    #SCOPE, @#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1728 004722 012737 000340 000022          MOV    #340, @#IOTVEC+2 ;;LEVEL 7
1729 004730 012737 017624 000030          MOV    #ERROR, @#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1730 004736 012737 000340 000032          MOV    #340, @#EMTVEC+2 ;;LEVEL 7
1731 004744 012737 023104 000034          MOV    #STRAP, @#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1732 004752 012737 000340 000036          MOV    #340, @#TRAPVEC+2; LEVEL 7
1733 004760 012737 176543 023556          MOV    #176543, $HINUM ;;PRIME THE RANDOM NUMBER GENERATOR
1734 004766 012737 123456 023560          MOV    #123456, $LONUM ;;BOTH HIGH AND LOW WORDS
1735 004774 005037 001204          CLR    $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
1736 005000 005037 001206          CLR    $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1737 005004 112737 000001 001115          MOV    #1, $ERMAX ;;ALLOW ONE ERROR PER TEST
1738 005012 012737 005012 001106          MOV    #., $LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1739 005020 012737 005020 001110          MOV    #., $LPERR ;;SETUP THE ERROR LOOP ADDRESS
1740          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1741          ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1742 005026 013746 000004          MOV    @#ERRVEC, -(SP) ;;SAVE ERROR VECTOR
1743 005032 012737 005066 000004          MOV    #64$, @#ERRVEC ;;SET UP ERROR VECTOR
1744 005040 012737 177570 001140          MOV    #DSWR, SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
1745 005046 012737 177570 001142          MOV    #DISP, DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1746 005054 022777 177777 174056          CMP    #-1, @SWR ;;TRY TO REFERENCE HARDWARE SWR
1747 005062 001012          BNE    66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1748          ;;AND THE HARDWARE SWR IS NOT = -1
1749          BR    65$ ;;BRANCH IF NO TIMEOUT
1750 005066 012716 005074          64$: MOV    #65$, (SP) ;;SET UP FOR TRAP RETURN
1751 005072 000002          RTI
1752 005074 012737 000176 001140          65$: MOV    #SWREG, SWR ;;POINT TO SOFTWARE SWR
1753 005102 012737 000174 001142          MOV    #DISPREG, DISPLAY
1754 005110 012637 000004          66$: MOV    (SP)+, @#ERRVEC ;;RESTORE ERROR VECTOR
1755
1756 005114 012700 001160          MOV    #SREGAD, RO ;;FIRST ADDRESS
1757 005120 005020          1$: CLR    (RO)+ ;;CLEAR VARIABLE STORAGE
1758 005122 022700 001210          CMP    #SBELL, RO ;;DONE?
1759 005126 001374          BNE    1$ ;;NO--BRANCH
1760 005130 013737 001414 001150          MOV    @#TPS, @#STPS ;;SETUP THE STATUS AND BUFFER REG'S
    
```


1761	005136	013737	001416	001152		MOV	2#TPB,2#STPB	FOR THE TYPE ROUTINE
1762	005144	005227	177777			INC	2-1	FIRST START ?
1763	005150	001026				BNE	3\$	BR IF NOT
1764	005152	104401	047662			TYPE	TITLE	TYPE THE PROGRAM'S TITLE
1765	005156	005737	000042			TST	42	AUTO ACCEPT OR CHAIN MODE ?
1766	005162	001006				BNE	2\$	BR IF EITHER
1767	005164	122737	000011	000041		CMPB	2#11,41	LOADED FROM AN RPO4/5/6 ?
1768	005172	001002				BNE	2\$	BR IF NOT
1769	005174	104401	047762			TYPE	,LOADRV	INSTRUCT THE OPERATOR TO REMOVE THE PACK
1770								ON DRIVE 0 IF DRIVE 0 IS TO BE TESTED
1771	005200	105737	000041		2\$:	TSTB	2#41	LOADED FROM PAPER TAPE ?
1772	005204	001410				BEG	3\$	BR IF NOT
1773	005206	004737	050256			JSR	PC,\$SIZE	SIZE THE MEMORY
1774	005212	023727	050352	100000		CMP	\$LSTAD,#100000	16K OR MORE ON THE SYSTEM ?
1775	005220	103002				BHS	3\$	BR IF YES
1776	005222	104401	050160			TYPE	,NOLOAD	INFORM THE OPERATOR THAT THE 'XXDP' LOADER
1777								WILL BE OVERRITTEN
1778	005226	004737	021322		3\$:	JSR	PC,\$TKINT	TURN ON THE JTY KEYBOARD INTERRUPT
1779					.5BTTL	GET	VALUE FOR SOFTWARE SWITCH REGISTER	
1780	005232	005737	000042			TST	2#42	ARE WE RUNNING UNDER XXDP/ACT?
1781	005236	001006				BNE	67\$	BRANCH IF YES
1782	005240	023727	001140	000176		CMP	SMR,#SMREG	SOFTWARE SWITCH REG SELECTED?
1783	005246	001005				BNE	68\$	BRANCH IF NO
1784	005250	104406				GTSWR		GET SOFT-SWR SETTINGS
1785	005252	000403				BR	68\$	
1786	005254	112737	000001	001134	67\$:	MOVB	2#1,\$AUTOB	SET AUTO-MODE INDICATOR
1787	005262				68\$:			
1788	005262	005227	177777			INC	2-1	SEE IF FIRST START
1789	005266	001002				BNE	SRTINT	BR IF NOT
1790	005270	004737	050354			JSR	PC,GETADR	GET OR CHECK THE RH11 ADDRESS
1791	005274	104401	001215		SRTINT:	TYPE	,SCLF	CR-LF
1792	005300	004737	024002			JSR	PC,2#LP.AVL	CHECK FOR A LINE PRINTER
1793	005304	005037	177776			CLR	2#PS	ENSURE THE PRIORITY = 0
1794	005310	012737	000001	001104		MOV	2#1,\$ICNT	SET ITERATION COUNT TO 1
1795	005316	004737	031374			JSR	PC,2#GETSWR	GO CHECK FOR CONTROL SWITCHES
1796	005322	004737	024044			JSR	PC,2#ST.CLK	INITIALIZE THE CLOCK
1797	005326	004737	034502		SETVEC:	JSR	PC,RPINIT	CHECK THE DRIVE STATUS
1798	005332	012737	177777	034424		MOV	2-1,\$SAVEFG	SET THE SAVE REGISTERS FLAG
1799	005340	062727	177777	000000		ADD	2-1,\$0	FIRST START ?
1800	005346	103003				BCC	11\$	BR IF YES
1801	005350	005737	001224			TST	CNTRLC	CONTROL 'C' SWITCH SET ?
1802	005354	001102				BNE	SRTDRV	CONTINUE IF YES
1803	005356	012737	000340	177776	11\$:	MOV	2#PR7,\$PS	SET PRIORITY TO 7
1804	005364	005004				CLR	R4	DRIVE TABLE POINTER
1805	005366	104401	001215			TYPE	,SCLF	CR-LF
1806	005370	104401	043050			TYPE	,SYSTAT	TYPE STATUS HEADING
1807	005370				1\$:			
1808	005370	010446				MOV	R4,-(SP)	SAVE R4 FOR TYPEOUT
1809								TYPE DRIVE NUMBER
1810	005400	104403				TYPOS		GO TYPE--OCTAL ASCII
1811	005402	002				.BYTE	2	TYPE 2 DIGIT(S)
1812	005403	000				.BYTE	0	SUPPRESS LEADING ZEROS
1813	005404	104401	044010			TYPE	,MSG.SP	SPACES
1814	005410	104401	044010			TYPE	,MSG.SP	SPACES
1815	005410	105764	034336			TSTB	DRVSTA(R4)	CHECK DRIVE'S STATUS
1816	005420	100416				BMI	4\$	BR IF UNSAFE

1817	005422	001020			BNE	5\$:BR IF ONLINE
1818	005424	105764	034346		TSTB	DRVTP(R4)		:SEE IF OFFLINE OR NONEXISTENT
1819	005430	001404			BEQ	2\$:BR IF NONEXISTENT
1820	005432	100006			BPL	3\$:BR IF OFFLINE
1821	005434	104401	043144		TYPE	NOTRP		:DRIVE NOT AN RPO4/5/6
1822	005440	000440			BR	9\$:CHECK NEXT DRIVE
1823	005442	104401	043117	2\$:	TYPE	NOTPRS		:DRIVE NOT PRESENT
1824	005446	000435			BR	9\$:CHECK NEXT DRIVE
1825	005450	104401	043076	3\$:	TYPE	UNTOFF		:DRIVE OFFLINE
1826	005454	000405			BR	6\$:PRINT DRIVE TYPE
1827	005456	104401	043134	4\$:	TYPE	NOTSAF		:DRIVE UNSAFE
1828	005462	000402			BR	6\$:PRINT DRIVE TYPE
1829	005464	104401	043107	5\$:	TYPE	UNTON		:DRIVE ONLINE
1830	005470	104401	044010	6\$:	TYPE	MSG.SP		:SPACES
1831	005474	012737	043162	005540	MOV	#RPO4,8\$:ADDRESS OF RPO4 MESSAGE
1832	005502	132764	000001	034346	BITB	#BIT00,DRVTP(R4)		:RPO4 ?
1833	005510	001012			BNE	7\$:BR IF YES
1834	005512	012737	043167	005540	MOV	#RPO5,8\$:ADDRESS OF RPO5 MESSAGE
1835	005520	132764	000002	034346	BITB	#BIT01,DRVTP(R4)		:RPO5 ?
1836	005526	001003			BNE	7\$:BR IF YES
1837	005530	012737	043174	005540	MOV	#RPO6,8\$:ADDRESS OF RPO6 MESSAGE
1838	005536	104401		7\$:	TYPE			:TYPE THE DRIVE TYPE MESSAGE
1839	005540	000000		8\$:	.WORD	0		:MESSAGE ADDRESS HERE
1840	005542	104401	001215	9\$:	TYPE	SCR LF		:CR-LF
1841	005546	005204			INC	R4		:INCREMENT DRIVE NUMBER/TABLE POINTER
1842	005550	020427	000010		CMP	R4,#8.		:FINISHED ?
1843	005554	001310			BNE	1\$:BR IF NOT
1844	005556	005037	177775		CLR	PS		:SET PRIORITY BACK TO '0'
1845	005562	005737	001224	SRTDRV:	TST	#CNTRLC		:CONTROL "C" START/RESTART?
1846	005566	001417			BEQ	1\$:NO--BRANCH
1847	005570	013746	001222		MOV	SAVCSM,-(SP)		:GET THE PREVIOUS 'C.SMR' CONTENTS
1848	005574	063716	001220		ADD	C.SMR,(SP)		:SET UP TO SEE IF 'BIT00' IS DIFFERENT
1849	005600	032726	000001		BIT	#BIT00,(SP)+		:IS 'BIT00' DIFFERENT ?
1850	005604	001405			BEQ	9\$:BR IF NOT
1851	005606	013737	001220	001222	MOV	C.SMR,SAVCSM		:STORE PRESENT 'C.SMR' VALUE
1852	005614	004737	024322		JSR	PC,LODFLT		:RESET PARAMETERS TO THEIR DEFAULT VALUES
1853	005620	004737	031624	9\$:	JSR	PC,#GT.PRM		:GET PARAMETERS
1854	005624	000420			BR	4\$		
1855	005626	004737	024322	1\$:	JSR	PC,LODFLT		:SETUP DEFAULT PARAMETERS
1856	005632	005037	001232		CLR	DRVSEL		:NO DRIVES SELECTED
1857	005636	005000			CLR	RD		:DETERMINE THE DRIVES THAT
1858	005640	012701	000001		MOV	#1,R1		:ARE AVAILABLE FOR TESTING
1859	005644	105760	034336	2\$:	TSTB	DRVSTA(RD)		
1860	005650	003403			BLE	3\$		
1861	005652	156037	034452	001232	BISB	ATABIT(RD),#DRVSEL		
1862	005660	005200		3\$:	INC	RD		
1863	005662	106301			R1	R1		
1864	005664	001367			BNE	2\$		
1865	005666	005037	034426	4\$:	CLR	#SEEKFG		:CLEAR SEEK FLAG
1866	005672	032737	000400	001220	BIT	#SW08,#C.SMR		:DO SEEK BEFORE DATA TRANSFER?
1867	005700	001002			BNE	5\$:YES--BRANCH
1868	005702	005137	034426		COM	#SEEKFG		:NO
1869	005706	122737	000011	000041	5\$:	CMPB	#11,41 ;LOADED	:FROM AN RPO4/5/6 ?
1870	005714	001003			BNE	10\$:BR IF NOT
1871	005716	042737	000001	001232	BIC	#BIT00,DRVSEL		:CLEAR THE DRIVE 0 SELECTION BIT
1872	005724	104401	043201	10\$:	TYPE	,DRIVES		: 'DRIVES(S) TO BE TESTED'

M03

1873	005730	005037	017540		CLR	2#SENDCT	: DETERMINE PASSES TO MAKE AND
1874	005734	005000			CLR	RD	: THE DRIVES TO BE TESTED
1875	005736	013701	001232		MOV	2#DRVSEL,R1	: ANY DRIVES SELECTED?
1876	005742	001004			BNE	6S	: YES--BRANCH
1877	005744	104401	043232		TYPE	,NONE	: 'NONE'
1878							
1879	005750	000137	017356		JMP	2#SEOP	: GO TO END OF PROGRAM
1880	005754	006201		6S:	ASR	R1	: REPORT THE DRIVES TO BE TESTED
1881	005756	103011			BCC	7S	
1882	005760	005237	017540		INC	2#SENDCT	: GIVE THIS DRIVE A PASS
1883	005764	010046			MOV	RD,-(SP)	: SAVE RD FOR TYPEOUT
1884	005766	104403			TYPOS		: GO TYPE--OCTAL ASCII
1885	005770	001			.BYTE	1	: TYPE 1 DIGIT(S)
1886	005771	000			.BYTE	0	: SUPPRESS LEADING ZEROS
1887	005772	005701			TST	R1	: MORE DRIVES?
1888	005774	001404			BEQ	8S	: NO--BRANCH
1889	005776	104401	043237		TYPE	,COMMA	:
1890	006002	005200		7S:	INC	RD	: FORM DRIVE NUMBER
1891	006004	000763			BR	6S	
1892	006006	013737	017540	017532	MOV	2#SENDCT,2#SEOPCT	
1893	006014	005737	001244		TST	2#CLKSTA	: KW11-P AVAILABLE
1894	006020	003006			BGT	RSTRT1	: YES--BRANCH
1895	006022	032737	036000	001234	BIT	#36000,2#TSTNMS	: NO--ANY TIMING TESTS TO BE PERFORMED?
1896	006030	001402			BEQ	RSTRT1	: NO--BRANCH
1897	006032	104401	043241		TYPE	,NOCLOCK	: 'NO KW11-P CLOCK. TIMING TESTS WILL NOT BE PERFORMED'
1898	006036	005737	001232		RSTRT1: TST	DRVSEL	: ANY DRIVES SELECTED ?
1899	006042	001002			BNE	1S	: BR IF YES
1900	006044	000137	004560		JMP	START2	: GET DRIVE SELECTION ENTRY
1901	006050	005037	001254		1S: CLR	CHKDRV	: INIT. THE CHECK DRIVE KEY
1902	006054	012737	000001	001256	MOV	#1,DRVMSK	: START TO CHECK DESIRED DRIVES
1903	006062	033737	001256	001232	RSTRT2: BIT	DRVMSK,DRVSEL	: IS THIS DRIVE SELECTED?
1904	006070	001010			BNE	DRVOK	: YES--GO CHECK IF DRIVE IS READY FOR TESTING
1905	006072	012706	001100		RESTART: MOV	#STACK,SP	: SETUP THE STACK POINTER
1906	006076	005237	001254		INC	CHKDRV	: MOVE TO NEXT DRIVE NUMBER
1907	006102	106337	001256		ASLB	DRVMSK	: POSITION THE MASK
1908	006106	103753			BCS	RSTRT1	: BRANCH IF THE DRIVE NUMBER NEEDS INITIALIZED
1909	006110	000764			BR	RSTRT2	
1910							
1911	006112	013702	001254		DRVOK: MOV	CHKDRV,R2	: PICKUP THE DRIVE NUMBER
1912	006116	105762	034336		TSTB	DRVSTA(R2)	: IS DESIRED DRIVE ON-LINE?
1913	006122	003005			BGT	1S	: YES, BRANCH
1914	006124	104011			ERROR	11	: DRIVE SELECTED IS NOT ONLINE
1915	006126	043737	001256	001232	BIC	DRVMSK,DRVSEL	: CLEAR DRIVE'S SELECTION BIT
1916	006134	000756			BR	RESTART	: RETURN
1917	006136	010237	004104		1S: MOV	R2,2#DPB.A	: SET THE DRIVE NUMBER INTO THE DPB'S
1918	006142	010237	004124		MOV	R2,2#DPB.B	
1919	006146	010237	004144		MOV	R2,2#DPB.C	
1920	006152	010237	004164		MOV	R2,2#DTADPB	
1921	006156	004737	024736		JSR	PC,2#LDCMD	: LOAD COMMAND INTO DPB.B AND DPB.C
1922	006162	012737	017356	001252	MOV	#SEOP,2#BYPASS	: IF ERROR GO TO END OF PROGRAM
1923	006170	112737	000020	004105	MOV	#FMT22/256,DPB.A+1	: ASSUME 16 BIT FORMAT
1924	006176	032737	000001	001220	BIT	#BIT00,C.SWR	: 16 BIT FORMAT REQUESTED ?
1925	006204	001402			BEQ	2S	: BR IF YES
1926	006206	105037	004105		CLRB	DPB.A+1	: CLEAR THE 'FMT22' BIT
1927	006212	112737	000143	004106	2S: MOV	#SETFORM,DPB.A+2	: SET THE FORMAT BIT PER DPB.A+1
1928	006220	004037	025002		JSR	RD,2#CALL.A	: GO EXECUTE THE COMMAND

N03

MD-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 40
 DZRJA.014 19-APR-76 00:00 GET VALUE FOR SOFTWARE SWITCH REGISTER

1929	006224	112737	000107	004106	MOVB	#RECAL, @#DPB.A+2	:RECAL=COMMAND
1930	006232	004037	025002		JSR	RC, @CALL.A	:GO EXECUTE THE COMMAND
1931	006236	104401	043327		TYPE	TESTNG	: 'TESTING DRIVE'
1932	006242	010246			MOV	R2, -(SP)	:SAVE R2 FOR TYPEOUT
1933	006244	104403			TYPOS		:GO TYPE--OCTAL ASCII
1934	006246	001			.BYTE	1	:TYPE 1 DIGIT(S)
1935	006247	000			.BYTE	0	:SUPPRESS LEADING ZEROS
1936	006250	104401	044010		TYPE	,MSG.SP	:TYPE SPACES
1937	006254	104401	043351		TYPE	,SERIAL	: 'SERIAL NUMBER'
1938	006260	012700	000004		MOV	#4, R0	:FOUR DIGITS TO TYPE
1939	006264	013701	004234		MOV	RP, REG+30, R1	:SERIAL NUMBER
1940	006270	005002		3\$:	CLR	R2	:ZERO
1941	006272	006101			ROL	R1	:PUT THE NEXT DIGIT
1942	006274	006102			ROL	R2	:INTO R2
1943	006276	006101			ROL	R1	
1944	006300	006102			ROL	R2	
1945	006302	006101			ROL	R1	
1946	006304	006102			ROL	R2	
1947	006306	006101			ROL	R1	
1948	006310	006102			ROL	R2	
1949	006312	062702	000060		ADD	#'0, R2	:MAKE IT ASCII
1950	006316	010227			MOV	R2, (PC)+	:SAVE IT
1951	006320	000000		4\$:	.WORD	0	
1952	006322	104401	006320		TYPE	,4\$:TYPE
1953	006326	005300			DEC	R0	:ALL DIGITS TYPED?
1954	006330	003357			BGT	3\$:NO -- BRANCH
1955	006332	104401	001215		TYPE	,SCRLF	
1956	006336	113737	001364	001115	MOVB	ERR.CT, SERMAX	:SETUP MAX ERROR COUNT

2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068

006344
006344 000240
006346 033737 001424 001234
006354 001002
006356 000137 006506
006362 012737 000000 001102
006370 004737 024560
006374 012737 006470 001110
006402 013777 001102 172532
006410 013737 001506 001204
006416 112737 000031 001115
006424 112737 000107 004106
006432 113737 001524 004134
006440 113737 001516 004135
006446 013737 001512 004136
006454 012737 006504 001252
006462 012737 006470 001106
006470 012706 001100
006474 004037 025002
006500 004037 025114
006504 000004

```

: * THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE
: * COMMAND CYCLE AND THEN SEEK FORWARD TO CYLINDER "LC" AT
: * THE COMPLETION OF BOTH COMMANDS STATUS INDICATIONS ARE
: * CHECKED TO ENSURE NO ERRORS OCCURRED.
: *****
TST0:
NOP
BIT 2#BITS+(0*2),TSTNMS ;DO THIS TEST?
BNE 645 ;YES--BRANCH
JMP TST1 ;NO--GO TO THE NEXT TEST
MOV 645: 80,2#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LOOPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV 8TEST0,2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV 8STNM,2DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV 2#RPT,8TIMES ;GET THE ITERATION COUNT
MOVB 825,8SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOVB 2#RECAL,2#DPB.A+2 ;RECAL=COMMAND
MOVB 2#FS,2#DPB.B+10 ;FS
MOVB 2#FT,2#DPB.B+11 ;FT
MOV 2#LC,2#DPB.B+12 ;LC
MOV 8EXIT0,2#BYPASS ;GO TO EXIT0 ON ERROR
MOV 8TEST0,8LPADR ;SETUP LOOP ADDRESS
TEST0: MOV 8STACK,8P ;SET UP STACK POINTER
JSR 80,2#CALL.A ;GO EXECUTE THE COMMAND
JSR 80,2#CALL.B ;GO EXECUTE THE COMMAND
EXIT0: SCOPE ;LOOP
: *****
: #TEST 1 SEEK/SEEK TEST
: * THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
: * CYCLE TO "LC", "LT", "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
: * "FC", "FT", "FS". AT THE COMPLETION OF EACH SEEK, THE PROPER
: * INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
: * "LC" WILL DEFAULT TO 128 AND "FC", "FT", "LT", "FS", AND "LS"
: * WILL DEFAULT TO 0
: *****
TST1:
NOP
BIT 2#BITS+(1*2),TSTNMS ;DO THIS TEST?
BNE 645 ;YES--BRANCH
JMP TST2 ;NO--GO TO THE NEXT TEST
MOV 645: 81,2#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LOOPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV 8TEST1,2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV 8STNM,2DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV 2#RPT,8TIMES ;GET THE ITERATION COUNT
MOVB 825,8SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOVB 2#FS,2#DPB.B+10 ;FS
MOVB 2#LS,2#DPB.C+10 ;LS
MOVB 2#FT,2#DPB.B+11 ;FT

```


2069 006610 113737 001520 004155
2070 006616 013737 001510 004136
2071 006624 013737 001512 004156
2072 006632 012737 006662 001252
2073 006640 012737 006646 001106
2074 006646 012706 001100
2075 006652 004037 025304
2076 006656 004037 025114
2077 006662 000004

MOV B @#T,@#DPB.C+11 ;LT
MOV @#FC,@#DPB.B+12 ;FC
MOV @#LC,@#DPB.C+12 ;LC
MOV @EXIT1,@#BYPASS ;GO TO EXIT1 ON ERROR
MOV @TEST1,@#SLPADR ;SETUP LOOP ADDRESS
TEST1: MOV @STACK,SP ;SET THE STACK POINTER
JSR RD,@#CALL.C ;GO EXECUTE THE COMMAND
JSR RD,@#CALL.B ;GO EXECUTE THE COMMAND
EXIT1: SCOPE ;LOOP

*TEST 2 INCREMENT/SEEK TEST

THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC". WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS "LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC" UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.

*TST2:

2092 006664
2093 006664 000240
2094 006666 033737 001430 001234
2095 006674 001002
2096 006676 000137 007102
2097 006702 012737 000002 001102
2098
2099 006710 004737 024560
2100 006714 012737 006774 001110
2101 006722 013777 001102 172212
2102 006730 013737 001506 001204
2103 006736 112737 000031 001115
2104 006744 012737 006752 001106
2105 006752 113737 001524 004134
2106 006760 113737 001516 004135
2107 006766 012737 007100 001252
2108 006774 013737 001510 004136
2109 007002 012737 007002 001110
2110 007010 012706 001100
2111 007014
2112 007014 004037 025114
2113 007020 063737 001514 004136
2114 007026 023737 001512 004136
2115 007034 002367
2116 007036 013737 001512 004136
2117 007044 012737 007044 001110
2118 007052 012706 001100
2119 007056
2120 007056 004037 025114
2121 007062 163737 001514 004136
2122 007070 023737 001510 004136
2123 007076 003767
2124 007100 000004

NOP
BIT @#BITS+(2*2),TSTNMS ;DO THIS TEST?
BNE 648 ;YES--BRANCH
JMP TST3 ;NO--GO TO THE NEXT TEST
648: MOV @2,@#TSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LOOPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV @TEST2,@#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV @TSTNM,@#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV @#RPT,@#TIMES ;GET THE ITERATION COUNT
MOV @25,@#SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOV @#S,@#SLPADR ;SETUP LOOP ADDRESS
18: MOV @#FS,@#DPB.B+10 ;FS
MOV @#FT,@#DPB.B+11 ;FT
MOV @EXIT2,@#BYPASS ;GO TO EXIT2 ON ERROR
TEST2: MOV @#FC,@#DPB.B+12 ;FC
MOV @#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV @#STACK,SP ;LOAD THE STACK POINTER
INCSK: JSR RD,@#CALL.B ;GO EXECUTE THE COMMAND
ADD @#IC,@#DPB.B+12 ;MOVE TO NEXT CYLINDER
CMP @#LC,@#DPB.B+12 ;OUT OF CYLINDERS?
BGE INCSK ;NO--BRANCH
MOV @#LC,@#DPB.B+12
MOV @#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV @#STACK,SP ;LOAD THE STACK POINTER
DECSK: JSR RD,@#CALL.B ;GO EXECUTE THE COMMAND
SUB @#IC,@#DPB.B+12
CMP @#FC,@#DPB.B+12
BLE DECSK
EXIT2: SCOPE ;LOOP

E04

2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180

007102
007102 000240
007104 033737 001432 001234
007112 001002
007114 000137 007276
007120 012737 000003 001102
007126 004737 024560
007132 012737 007212 001110
007140 013777 001102 171774
007146 013737 001506 001204
007154 112737 000031 001115
007162 012737 007170 001106
007170 113737 001524 004134
007176 113737 001516 004135
007204 012737 007274 001252
007212 013737 001510 004136
007220 012737 007220 001110
007226 012706 001100
007232 004037 025114
007236 013701 001514
007242 012737 007242 001110
007250 012706 001100
007254 010137 004136
007260 004037 025114
007264 006301
007266 020137 001512
007272 003770
007274 000004

:TEST 3 STEPPING SEEK TEST

:* THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4,
8,16,32,64,128, AND 256. AT THE COMPLETION OF EACH SEEK
COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER
OPERATION.

TST3:

NOP
BIT 2#BITS+(3*2),TSTNMS ;DO THIS TEST?
BNE 645 ;YES--BRANCH
JMP TST4 ;NO--GO TO THE NEXT TEST
MOV #3,2#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV 2#TST3,2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV 2#STSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV 2#RPT,\$TIMES ;GET THE ITERATION COUNT
MOVB #25,\$SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOV #15,\$LPADR ;SETUP TEST LOOP ADDRESS
IS: MOVB 2#FS,2#DPB.B+10 ;FS
MOVB 2#FT,2#DPB.B+11 ;FT
MOV 2#EXIT3,2#BYPASS ;GO TO BYPASS ON ERROR
TEST3: MOV 2#FC,2#DPB.B+12 ;FC
MOV 2#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV 2#STACK,SP ;LOAD THE STACK POINTER
JSR RO,2#CALL.B ;GO EXECUTE THE COMMAND
MOV IC,R1 ;CYLINDER 1
MOV 2#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV 2#STACK,SP ;LOAD THE STACK POINTER
IS: MOV R1,2#DPB.B+12 ;DESIRED CYLINDER
JSR RO,2#CALL.B ;GO EXECUTE THE COMMAND
ASL R1 ;MOVE TO NEXT CYLINDER
CMP R1,2#LC ;DONE?
BLE IS ;NO--LOOP
EXIT3: SCOPE

:TEST 4 OSCILLATING SEEK TEST

:* THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK
TO "FC". "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER
"LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC". AT THE
COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE
EXAMINED TO ENSURE PROPER OPERATION.

TST4:

NOP
BIT 2#BITS+(4*2),TSTNMS ;DO THIS TEST?
BNE 645 ;YES--BRANCH
JMP TST5 ;NO--GO TO THE NEXT TEST
MOV #4,2#STSTNM ;SET UP TEST NUMBER AND

2237 007650 163701 001514
2238 007654 020137 001510
2239 007660 002340
2240 007662 000004

6S: SUB @#IC,R1 ;NEXT CYLINDER
CMP R1,@#FC ;DONE?
BGE #5 ;NO--BRANCH
EXIT4: SCOPE ;LOOP

;#TEST 5 CONVERGING/DIVERGING SEEK TEST

;; THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
;; SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
;; BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
;; GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
;; LESS THAN THE INITIAL VALUE OF "NC1". AT THE COMPLETION OF
;; EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
;; ENSURE PROPER OPERATION. "NC1" AND "NC2" DEFAULT TO
;; "FC" AND "LC" RESPECTIVELY.

TST5:

007664
2256 007664 000240
2257 007666 033737 001436 001234
2258 007674 001002
2259 007676 000137 010064
2260 007702 012737 000005 001102
2261
2262 007710 004737 024560
2263 007714 012737 007774 001110
2264 007722 013777 001102 171212
2265 007730 013737 001506 001204
2266 007736 112737 000031 001115
2267 007744 012737 007752 001106
2268 007752 113737 001524 004134 1S:
2269 007760 113737 001516 004135
2270 007766 012737 010062 001252
2271 007774 013701 001510
2272 010000 013702 001512
2273 010004 012737 010004 001110
2274 010012 012706 001100
2275 010016 010137 004136 1S:
2276 010022 004037 025114
2277 010026 010237 004136
2278 010032 004037 025114
2279 010036 063701 001514
2280 010042 163702 001514
2281 010046 020137 001512
2282 010052 003003
2283 010054 020237 001510
2284 010060 002356
2285 010062 000004

NOP ;DO THIS TEST?
BIT @#BITS+(5*2),TSTNMS ;YES--BRANCH
BNE #4S ;NO--GO TO THE NEXT TEST
JMP TST6 ;SET UP TEST NUMBER AND
64S: MOV #5,@#TSTN ;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV @#TST5,@#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV @#TSTN,@#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV @#RPT,@#STRES ;GET THE ITERATION COUNT
MOVB #25,@#SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOV #15,@#SLPADR ;SETUP LOOP ADDRESS
1S: MOVB @#FS,@#DPB.B+10 ;FS
MOVB @#FT,@#DPB.B+11 ;FT
MOV @#EXITS,@#BYPASS ;GO TO EXITS ON ERROR
TEST5: MOV @#FC,R1 ;START NC1 AT FC
MOV @#LC,R2 ;START NC2 AT LC
MOV #,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV @#STACK,SP ;LOAD THE STACK POINTER
1S: MOV R1,@#DPB.B+12 ;NC1
JSR R0,@#CALL.B ;GO EXECUTE THE COMMAND
MOV R2,@#DPB.B+12 ;NC2
JSR R0,@#CALL.B ;GO EXECUTE THE COMMAND
ADD @#IC,R1 ;NEXT NC1
SUB @#IC,R2 ;NEXT NC2
CMP R1,@#LC ;DONE?
BGT EXITS ;YES--BRANCH
CMP R2,@#FC ;?
BGE #S ;NO--BRANCH
EXIT5: SCOPE ;LOOP

;#TEST 6 SERVO ADDRESSING LOGIC NOISE GENERATOR

;; IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
;; NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5. NOW "NC" IS UPDATED
;; BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNTIL "LC" IS

2286
2287
2288
2289
2290
2291
2292

H04

2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348

010064
010064 000240
010066 033737 001440 001234
010074 001002
010076 000137 010330
010102 012737 000006 001102
010110 004737 024560
010114 012737 010174 001110
010122 013777 001102 171012
010130 013737 001506 001204
010136 112737 000031 001115
010144 012737 010152 001106
010152 113737 001524 004134
010160 113737 001516 004135
010166 012737 010326 001252
010174 013701 001510
010200 013702 001512
010204 162702 000005
010210 012737 010210 001110
010216 012706 001100
010222 020102
010224 003040
010226 010137 004136
010232 004037 025114
010236 062737 000004 004136
010244 004037 025114
010250 162737 000003 004136
010256 004037 025114
010262 062737 000002 004136
010270 004037 025114
010274 162737 000001 004136
010302 004037 025114
010306 062737 000003 004136
010314 004037 025114
010320 063701 001514
010324 000736
010326 000004

```

;* EXCEEDED BY ANY OF THE ABOVE VALUES. THE INITIAL VALUE OF "NC"
;* IS "FC". AT THE COMPLETION OF EACH SEEK COMMAND THE
;* PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
*****
TST6:
NOP
BIT 2#BITS+(6*2),TSTNMS ;DO THIS TEST?
BNE 64S ;YES--BRANCH
JMP TST7 ;NO--GO TO THE NEXT TEST
64S: MOV #6,2#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LOOPM ;LOAD THE PARAMETERS FOR THE TEST
MOV 2#TEST6,2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV 2#STSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV 2#RPT,2#TIMES ;GET THE ITERATION COUNT
MOVB #25,SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOV #18,SLPADR ;SETUP LOOP ADDRESS
1S: MOVB 2#FS,2#DPB.B+10 ;FS
MOVB 2#FT,2#DPB.B+11 ;FT
MOV #EXIT6,2#BYPASS ;GO TO EXIT6 ON ERROR
TEST6: MOV 2#FC,R1 ;PICKUP "FC"
MOV 2#LC,R2 ;FORM LAST CYLINDER THAT
SUB #5,R2 ;IS AVAILABLE FOR TESTING
MOV #,SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV #STACK,SP ;LOAD THE STACK POINTER
1S: CMP R1,R2 ;LAST CYLINDER
BGT EXIT6 ;YES--BRANCH
MOV R1,2#DPB.B+12 ;NC
JSR RD,2#CALL.B ;GO EXECUTE THE COMMAND
ADD #4,2#DPB.B+12 ;NC+4
JSR RD,2#CALL.B ;GO EXECUTE THE COMMAND
SUB #3,2#DPB.B+12 ;NC+1
JSR RD,2#CALL.B ;GO EXECUTE THE COMMAND
ADD #2,2#DPB.B+12 ;NC+3
JSR RD,2#CALL.B ;GO EXECUTE THE COMMAND
SUB #1,2#DPB.B+12 ;NC+2
JSR RD,2#CALL.B ;GO EXECUTE THE COMMAND
ADD #3,2#DPB.B+12 ;NC+5
JSR RD,2#CALL.B ;GO EXECUTE THE COMMAND
ADD #1C,R1
BR 1S
EXIT6: SCOPE ;LOOP

```

```

*****
;TEST 7 RANDOM SEEK TEST
;* THIS TEST PERFORMS RANDOM SEEK OPERATIONS BETWEEN CYLINDERS 'FC'
;* 'LC'. AFTER EACH SEEK, THE POSITION OF THE DRIVE IS VERIFIED BY
;* READING A SECTOR FROM THE CURRENTLY ADDRESSED CYLINDER AND TRACK.
;* THE TRACK ADDRESS IS INCREMENTED FOR EACH SEEK SO THAT VERIFICATION
;* OF POSITIONING OCCURS USING EACH HEAD. TRACK ADDRESSES ARE INCREMENTED
;* BETWEEN PARAMETERS 'FT' AND 'LT'.
*****
TST7:

```

010330

2349	010330	000240				NOP		
2350	010332	033737	001442	001234		BIT	2#BITS+(7*2),TSTNMS ;DO THIS TEST?	
2351	010340	001002				BNE	64\$:YES--BRANCH
2352	010342	000137	010710			JMP	TST10	:NO--GO TO THE NEXT TEST
2353	010346	012737	000007	001102	64\$:	MOV	#7,2#STSTNM	:SET UP TEST NUMBER AND
2354								:CLEAR THE ERROR FLAG (SERFLG)
2355	010354	004737	024560			JSR	PC,LODPRM	:LOAD THE PARAMETERS FOR THE TEST
2356	010360	012737	010452	001110		MOV	#TEST7,2#SLPERR	:SETUP THE LOOP ON ERROR ADDRESS
2357	010366	013777	001102	170546		MOV	\$TSTNM,2\$DISPLAY	:LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2358	010374	013737	001506	001204		MOV	2#RPT,\$TIMES	:GET THE ITERATION COUNT
2359	010402	112737	000031	001115		MOV#B	#25,\$ERMAX	:MAX ERRORS ALLOWED FOR TEST
2360	010410	113737	001516	004135		MOV#B	FT,DPB.B+11	:LOAD STARTING TRACK ADDRESS
2361	010416	112737	000105	004106		MOV#B	#SEEK,2#DPB.A+2	:SEEK=COMMAND
2362	010424	112737	000173	004126		MOV#B	#READHD,DPB.B+2	:READ HEADER & DATA COMMAND
2363	010432	013704	034464			MOV	RPADR,R4	:UNIBUS ADDRESS OF THE RH11
2364	010436	012737	010706	001252		MOV	#EXIT7,BYPASS	:ERROR TERMINATION ADDRESS
2365	010444	012737	010452	001106		MOV	#TEST7,\$LPADR	:SETUP THE LOOP ON TEST ADDRESS
2366	010452	012706	001100		TEST7:	MOV	#STACK,SP	:SETUP THE STACK POINTER
2367	010456	013737	001510	004136		MOV	FC,DPB.B+12	:INITIAL CYLINDER ADDRESS
2368	010464	023737	001510	001512		CMP	FC,LC	:CYLINDER LIMITS THE SAME ?
2369	010472	001422				BEQ	1\$:BR IF THEY ARE
2370	010474	004737	023460			JSR	PC,\$RAND	:CYCLE THE RANDOM NUMBER GENERATOR
2371	010500	013746	023556			MOV	\$HNUM,-(SP)	:USE THE HIGH RANDOM NUMBER
2372	010504	005046				CLR	-(SP)	:UPPER DIVIDEND
2373	010506	013746	001512			MOV	LC,-(SP)	:FORM THE DIVISOR
2374	010512	005216				INC	(SP)	:INCREMENT
2375	010514	163716	001510			SUB	FC,(SP)	:SUBTRACT THE LOWER LIMIT
2376	010520	004737	023562			JSR	PC,\$DIV	:DIVIDE
2377	010524	062637	004136			ADD	(SP)+,DPB.B+12	:ADD THE REMAINDER TO THE INITIAL CYLINDER
2378	010530	005726				TST	(SP)+	:DISCARD THE QUOTIENT
2379	010532	013737	004136	004116		MOV	DPB.B+12,DPB.A+12	:COPY NEW CYLINDER ADDRESS
2380	010540				1\$:			
2381	010540	012737	010540	001110		MOV	#,\$SLPERR	:SETUP THE ERROR LOOP ADDRESS
2382	010546	012706	001100			MOV	#STACK,SP	:LOAD THE STACK POINTER
2383	010552	004037	025002			JSR	RD,2\$CALL.A	:GO EXECUTE THE COMMAND
2384	010556	012737	010556	001110		MOV	#,\$SLPERR	:SETUP THE ERROR LOOP ADDRESS
2385	010564	012706	001100			MOV	#STACK,SP	:LOAD THE STACK POINTER
2386	010570	113764	004104	000010		MOV#B	DPB.A,RPCS2(R4)	:SELECT THE DRIVE
2387	010576	016446	000020			MOV	RPLA(R4),-(SP)	:GET THE LOOK AHEAD REGISTER
2388	010602	006316				ASL	(SP)	:ALIGN THE SECTOR ADDRESS
2389	010604	006316				ASL	(SP)	:ALIGN THE SECTOR ADDRESS
2390	010606	000316				SWAB	(SP)	:PUT ADDRESS IN LOWER BYTE
2391	010610	105766	000001			TSTB	1(SP)	:IN THE 1ST 20% OF SECTOR ?
2392	010614	001401				BEQ	2\$:BR IF YES
2393	010616	105216				INCB	(SP)	:INCREMENT THE SECTOR ADDRESS
2394	010620	105216			2\$:	INCB	(SP)	:INCREMENT THE SECTOR ADDRESS
2395	010622	112637	004174			MOV#B	(SP)+,DTADPB+10	:LOAD THE DPB
2396	010626	013746	001630			MOV	PRMLT+22,-(SP)	:PUT LAST SECTOR ADDRESS ON THE STACK
2397	010632	005216				INC	(SP)	:INCREMENT IT
2398	010634	122637	004174			CMP#B	(SP)+,DTADPB+10	:NEW SECTOR ADDRESS TOO LARGE ?
2399	010640	103007				BHIS	4\$:BR IF NOT
2400	010642	103403				BLO	3\$:BR IF ADDRESS IS 2 GREATER
2401	010644	105037	004174			CLRB	DTADPB+10	:RESET TO SECTOR ADDRESS 0
2402	010650	000403				BR	4\$:CONTINUE
2403	010652	112737	000001	004174	3\$:	MOV#B	#1,DTADPB+10	:RESET ADDRESS TO SECTOR 1
2404	010660				4\$:			


```

405 010660 004037 025114 JSR RD,%CALL.B ;GO EXECUTE THE COMMAND
406 010664 105237 004135 INCB DPB.B+11 ;INCREMENT THE TRACK ADDRESS
407 010670 123737 004135 001520 CMPB DPB.B+11,LT ;MAXIMUM ?
408 010676 101403 BLOS EXIT7 ;BR IF NOT
409 010700 113737 001516 004135 MOVB FT,DPB.B+11 ;RELOAD STARTING TRACK ADDRESS
410 010706 000004 EXIT7: SCOPE ;LOOP ?

```

```

*****
;TEST 10 SERVO SETTLE DOWN TEST

```

```

** THIS TEST VERIFIES THAT THE SERVO HAS SETTLED DOWN AND THAT
** THE DRIVE IS ON CYLINDER WHEN THE DRIVE INDICATES SEEK COMPLETE.
** RANDOM SEEKS ARE ISSUED BETWEEN CYLINDERS 'NC1' AND 'NC1+IC'
** ('NC1' STARTS AT VALUE 'FC'). AT THE COMPLETION OF 1000 (10) SEEKS,
** 'NC1' IS INCREMENTED BY VALUE 'IC' AND THE SEQUENCE IS REPEATED.
** THE TEST IS COMPLETED WHEN 'NC1' HAS BEEN INCREMENTED BEYOND 'LC'.
**
** WHEN THE SEEK COMPLETES, THE PROGRAM READS THE DRIVE'S LOOK-AHEAD
** REGISTER (RPLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING INTO
** POSITION. THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND
** FOR THAT SECTOR. IF THE DRIVE'S POSITIONER HAS NOT SETTLED DOWN OR
** IF THE POSITIONER IS NOT ON CYLINDER (IF THE DRIVE IS AN RPO4, THE
** OFF CYLINDER CONDITION MUST LAST FOR AT LEAST 800 US), THE DRIVE
** WILL REPORT A 'MRU' ERROR. (RPOS/6'S MAY ALSO REPORT 'NHS' ERROR UNDER
** ERRORS IN THIS TEST INDICATE THAT THE SERVO SYSTEM MAY NOT BE ADJUSTED
** CORRECTLY, THAT THE DRIVE IS MALFUNCTIONING, OR THAT A PACK WITH
** MARGINAL SERVO TRACKS IS MOUNTED ON THE DRIVE.
**
** THIS TEST USES THE EXTENTION BITS IN THE LOOK-AHEAD REGISTER TO DETERMINE
** WHETHER OR NOT IT CAN PICK UP THE SECTOR ROTATING INTO POSITION. THE
** TEST IS OPTIMIZED SUCH THAT IF THE DRIVE SIGNALS SEEK DONE WITHIN
** THE FIRST 80% OF THE SECTOR CURRENTLY UNDER THE HEAD, THE TEST WILL
** TRY TO ADDRESS THE NEXT SECTOR. BASED ON OBSERVATION, THE PROGRAM
** IS ABLE TO START THE OPERATION WITHOUT LOSING A REVOLUTION MOST OF
** THE TIME.
**
** THIS TEST IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW
** HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS. THE NECESSARY TIME
** TIME DEPENDENT PARAMETERS OCCUR FREQUENTLY ENOUGH WITHIN THE REQUIRED
** RANGE TO PERMIT THIS TEST TO BE EFFECTIVE.

```

```

*****
TST10:
010710 000240 NOP
010710 033737 001444 001234 BIT %BITS+<10*2>,TSTNMS ;DO THIS TEST?
010712 001002 BNE 64$ ;YES--BRANCH
010720 000137 012014 JMP TST11 ;NO--GO TO THE NEXT TEST
010722 000010 001102 64$: MOV #10,%STSTNM ;SET UP TEST NUMBER AND
010726 004737 024560 JSR PC,LODPRM ;CLEAR THE ERROR FLAG (SERFLG)
010734 001116 001110 MOV #TST10,%SLPERR ;LOAD THE PARAMETERS FOR THE TEST
010740 013777 001506 001204 MOV %STSTNM,%DISPLAY ;SETUP THE LOOP ON ERROR ADDRESS
010746 0112737 000031 001115 MOV %RPT,%TIMES ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
010754 010776 001106 001106 MOVB #25,%SERMAX ;GET THE ITERATION COUNT
010762 010776 001106 001106 MOV #18,%LPADR ;MAX ERRORS ALLOWED FOR TEST
010770 010776 001106 001106 ;SETUP THE LOOP ADDRESS

```


K04

761	010776				1S:	MOV	#SEEK, 2#DPB.A+2	:SEEK=COMMAND
762	010776	112737	000105	004106		MOV	#WRITE, DTADPB+2	:COMMAND
763	011004	112737	000161	004166		MOV	FT, DTADPB+11	:TRACK ADDRESS FOR THE WRITE
764	011012	113737	001516	004175		MOV	FC, DPB.A+12	:CYLINDER ADDRESS FOR THE SEEK
765	011020	013737	001510	004116		MOV	FC, DTADPB+12	:CYLINDER ADDRESS FOR THE WRITE
766	011026	013737	001510	004176		MOV	FC, NC1	:STARTING CYLINDER
767	011034	013737	001510	001532		MOV	IC, DELTA	:CYLINDER INCREMENT VALUE
768	011042	013737	001514	001350		MOV	#-<256, #4, >, DTADPB+4	:WORD COUNT
769	011050	012737	176000	004170		MOV	#BUFFER, DTADPB+6	:BUFFER ADDRESS
770	011056	012737	047662	004172		CLR	RO	:PATTERN POINTER (MC PATTERN)
771	011064	005000				JSR	PC, SETBUF	:LOAD THE WRITE BUFFER
772	011066	004737	030330			CLR	RI	:CLEAR REGISTER
773	011072	005001				MOV	DPB.A, R1	:LOAD DRIVE ADDRESS
774	011074	113701	004104			MOV	RPAOR, R4	:UNIBUS ADDRESS OF THE RH11
775	011100	013704	034464			JSR	PC, CLAQE	:CLEAR THE OPERATION QUEUES
776	011104	004737	042466			MOV	#EXIT10, BYPASS	:ERROR EXIT FROM TEST
777	011110	012737	012012	001252				
778	011116				TEST10:	MOV	#, SLPERR	:SETUP THE ERROR LOOP ADDRESS
779	011116	012737	011116	001110		MOV	#STACK, SP	:LOAD THE STACK POINTER
780	011124	012706	001100			MOV	#PR7, 2#PS	:SET PRIORITY TO 7
781	011130	012737	000340	177776		TST	CLKSTA	:SEE WHICH CLOCK ON SYSTEM
782	011136	005737	001244			BEQ	3S	:BR IF NO CLOCK
783	011142	001415				BMI	1S	:BR IF KW11-L CLOCK
784	011144	100405				MOV	2PKV, -(SP)	:SAVE THE VECTOR
785	011146	017746	170222			MOV	PKV, -(SP)	:SAVE THE VECTOR ADDRESS
786	011152	013746	001374			BR	2S	:CONTINUE
787	011156	000404				MOV	2LKV, -(SP)	:SAVE THE 'L' CLOCK VECTOR
788	011160	017746	170222		1S:	MOV	LKV, -(SP)	:SAVE THE VECTOR ADDRESS
789	011164	013746	001406		2S:	MOV	#TST10B, 2(SP)	:CHANGE THE VECTOR
790	011170	012776	011746	000000	3S:	MOV	#DORT1, 2RPVEC	:CHANGE THE RPO4/RPOS VECTOR
791	011176	012777	027022	003362		MOV	#8, SEKTR	:LOAD THE SEEK TIMER
792	011204	012737	000010	001344		MOV	#BIT05, RPCS2(R4)	:INIT THE MASSBUS
793	011212	012764	000040	000010		MOV	R1, RPCS2(R4)	:RESELECT THE DRIVE
794	011220	110164	000010			MOV	DPB.A+12, RPCA(R4)	:LOAD THE CYLINDER ADDRESS
795	011224	013764	004116	000034		MOV	DPB.A+12, CYL.DS	:CYLINDER ADDRESS FOR ERROR MESSAGE
796	011232	013737	034116	001270		MOV	#SEEK, RPCS1(R4)	:START THE SEEK
797	011240	112764	000105	000000		CLR	2#PS	:CLEAR THE PRIORITY
798	011246	005037	177776			TSTB	RPDS1(R4)	:HAS THE DRIVE FINISHED ?
799	011252	105764	000012		4S:	BMI	5S	:BR IF IT HAS
800	011256	100402				WAIT		:WAIT FOR THE OPERATION TO COMPLETE
801	011260	000001				BR	4S	:CONTINUE
802	011262	000773				MOV	#PR7, 2#PS	:CHANGE PRIORITY TO MAX
803	011264	012737	000340	177776	5S:	BIT	#BIT14, RPDS1(R4)	:ERROR ?
804	011272	032764	040000	000012		BEQ	6S	:BR IF NOT
805	011300	001412				MOV	#DPB.A, R2	:DPB POINTER
806	011302	012702	004104			JSR	PC, SVRH11	:SAVE THE REGISTERS
807	011306	004737	042004			ERROR	23	:ERROR DURING SEEK
808	011312	104023				MOV	#BIT05, RPCS2(R4)	:INIT THE MASSBUS
809	011314	012764	000040	000010		MOV	R1, RPCS2(R4)	:RESELECT THE DRIVE
810	011322	110164	000010			MOV	#ISR, 2RPVEC	:SETUP THE RPO4/RPOS VECTOR
811	011326	012777	037346	023132	6S:	TST	CLKSTA	:WHICH CLOCK
812	011334	005737	001244			BEQ	TST10A	:BR IF NONE
813	011340	001405				MOV	2(SP), 2(SP)	:RELOAD THE CLOCK VECTOR
814	011342	016676	000002	000000		ADD	#4, SP	:CORRECT THE STACK POINTER
815	011350	062706	000004					
816	011354				TST10A:			

2517	011354	012737	011354	001110	MOV	#,SLPERR	:SETUP THE ERROR LOOP ADDRESS
2518	011362	012706	001100		MOV	#STACK,SP	:LOAD THE STACK POINTER
2519	011366	110164	000010		MOVB	R1,RPCS2(R4)	:SELECT THE DRIVE
2520	011372	016446	000020		MOV	RP(A(R4),-(SP)	:GET THE LOOK AHEAD REGISTER
2521	011376	006316			ASL	(SP)	:ALIGN THE SECTOR ADDRESS
2522	011400	006316			ASL	(SP)	:ALIGN THE SECTOR ADDRESS
2523	011402	000316			SWAB	(SP)	:PUT ADDRESS IN LOWER BYTE
2524	011404	122766	000300	000001	CMPB	#300,1(SP)	:IN THE LAST 20% OR SECTOR ?
2525	011412	001001			BNE	2\$:BR IF NOT
2526	011414	105216			INCB	(SP)	:INCREMENT THE SECTOR ADDRESS
2527	011416	105216			INCB	(SP)	:INCREMENT THE SECTOR ADDRESS
2528	011420	112637	004174		MOVB	(SP)+,DTADPB+10	:LOAD THE DPB
2529	011424	013746	001630		MOV	PRMLT+22,-(SP)	:PUT MAXIMUM SECTOR ADDRESS ON THE STACK
2530	011430	005216			INC	(SP)	:INCREMENT PAST THE MAXIMUM ADDRESS
2531	011432	122637	004174		CMPB	(SP)+,DTADPB+10	:NEW SECTOR ADDRESS TOO LARGE ?
2532	011436	101007			BHI	4\$:BR IF NOT
2533	011440	103403			BLO	3\$:BR IF ADDRESS IS 2 GREATER THAN MAXIMUM
2534	011442	105037	004174		CLRB	DTADPB+10	:RESET TO SECTOR ADDRESS 0
2535	011446	000403			BR	4\$:CONTINUE
2536	011450	112737	000001	004174	MOVB	#1,DTADPB+10	:RESET ADDRESS TO SECTOR 1
2537	011456	012703	004170		MOV	#DTADPB+4,R3	:POINTER
2538	011462	012764	000111	000000	MOV	#DRVCLR,RPCS1(R4)	:CLEAR THE DRIVE
2539	011470	012364	000002		MOV	(R3)+,RPWC(R4)	:LOAD THE WORD COUNT
2540	011474	012364	000004		MOV	(R3)+,RPBA(R4)	:LOAD THE BUFFER ADDRESS
2541	011500	012364	000006		MOV	(R3)+,RPDA(R4)	:LOAD THE TRACK/SECTOR ADDR
2542	011504	005037	004202		CLR	DTADPB+16	:RESET 'DONE' INDICATOR
2543	011510	012737	004164	034376	MOV	#DTADPB,TRANSW	:LOAD 'TRANSFER' DPB ADDRESS
2544	011516	010137	034450		MOV	R1,DTUM	:ADDRESS OF DRIVE TRANSFERING
2545	011522	112761	000001	034326	MOVB	#1,DRVACT(R1)	:SET DRIVE ACTIVE INDICATOR
2546	011530	006301			ASL	R1	:SHIFT DRIVE ADDRESS
2547	011532	012761	001750	034430	MOV	#1000.,TIMER(R1)	:SETUP THE OPERATION TIMER
2548	011540	006201			ASR	R1	:RESTORE R1
2549	011542	013764	004166	000000	MOV	DTADPB+2,RPCS1(R4)	:START THE OPERATION
2550	011550	005037	177776		CLR	#PS	:CLEAR THE PRIORITY
2551	011554	004037	025514		JSR	RD,DRVCL1	:WAIT FOR OPERATION TO COMPLETE
2552	011560	023727	001346	001750	CMP	SEKCNT,#1000.	:FINISHED SEEKS ?
2553	011566	001026			BNE	6\$:BR IF NOT
2554	011570	005037	001346		CLR	SEKCNT	:CLEAR THE SEEK COUNT
2555	011574	063737	001514	001532	ADD	IC,NC1	:ADD THE INCREMENT
2556	011602	023737	001532	001512	CMP	NC1,LC	:EXCEEDED THE CYLINDER LIMIT ?
2557	011610	103100			BHIS	EXIT10	:BR IF IT HAS
2558	011612	013737	001512	001350	MOV	LC,DELTA	:GET THE NEXT 'ZONE' ADDRESS
2559	011620	163737	001532	001350	SUB	NC1,DELTA	:CHECK THE DIFFERENCE
2560	011626	023737	001514	001350	CMP	IC,DELTA	:DIFFERENCE GREATER THAN THE INCREMENT ?
2561	011634	101003			BHI	6\$:BR IF IT IS
2562	011636	013737	001514	001350	MOV	IC,DELTA	:USE THE ICREMENT PARAMETER
2563	011644	005237	001346		INC	SEKCNT	:COUNT THE NEXT SEEK
2564	011650	023737	001510	001512	CMP	FC,LC	:BEGINNING AND ENDING CYLINDERS THE SAME ?
2565	011656	001002			BNE	7\$:BR IF NOT
2566	011660	000137	011116		JMP	TEST10	:BR IF THEY ARE
2567	011664	013737	001532	004116	MOV	NC1,DPB.A+12	:RESET THE CYLINDER ADDRESS
2568	011672	004737	023460		JSR	PC,\$RAND	:CYCLE THE RANDOM NUMBER GENERATOR
2569	011676	013746	023556		MOV	SHNUM,-(SP)	:USE THE HIGH RANDOM NUMBER
2570	011702	005046			CLR	-(SP)	:CLEAR THE UPPER DIVIDEND
2571	011704	013746	001350		MOV	DELTA,-(SP)	:FORM THE DIVISOR
2572	011710	005216			INC	(SP)	:INCREMENT


```

2573 011712 004737 023562 JSR PC,SDIV ;DIVIDE
2574 011716 062637 004116 ADD (SP)+,DPB.A+12 ;ADD THE REMAINDER TO THE INITIAL CYLINDER
2575 011722 005726 TST (SP)+ ;DISCARD THE QUOTIENT
2576 011724 023737 004116 004176 CMP DPB.A+12,DTADPB+12 ;SAME CYLINDER SELECTED AS LAST TIME ?
2577 011732 001754 BEQ 75 ;BR IF IT WAS
2578 011734 013737 004116 004176 MOV DPB.A+12,DTADPB+12 ;COPY NEW CYLINDER ADDRESS
2579 011742 000137 011116 JMP TEST10 ;CONTINUE
2580 011746 005337 001344 TST10B: DEC SEKTR ;DECREMENT THE SEEK TIMER
2581 011752 001016 BNE 15 ;CONTINUE IF NOT DONE
2582 011754 012702 004104 MOV #DPB.A,R2 ;DPB ADDRESS
2583 011760 004737 042004 JSR PC,SVR411 ;SAVE THE REGISTERS
2584 011764 104024 ERROR 24 ;TIMEOUT DURING SEEK
2585 011766 012764 000040 000010 MOV #BIT05,RPCS2(R4) ;INIT THE MASSBUS
2586 011774 110164 000010 MOV R1,RPCS2(R4) ;RESELECT THE DRIVE
2587 012000 016676 000002 000000 MOV 2(SP),2(SP) ;RESTORE THE CLOCK VECTOR ADDRESS
2588 012006 000401 BR EXIT10 ;ABORT THE TEST
2589 012010 000002 15: RTI ;RETURN
2590 012012 000004 EXIT10: SCOPE ;LOOP ?

```

:TEST 11 ALL SEEKS TEST

THIS TEST VERIFIES THAT THE DISK DRIVE CAN SEEK FROM EACH CYLINDER TO ALL OTHER CYLINDERS.

BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER BETWEEN 'FC' AND 'LC' FROM CYLINDER 'FC'. THE BEGINNING CYLINDER ADDRESS IS INCREMENTED AND THE TEST SEEKS BETWEEN THE NEW CYLINDER ADDRESS AND ALL CYLINDERS BETWEEN 'FC' AND 'LC'. THE SEQUENCE CONTINUES UNTIL ALL CYLINDERS HAVE BEEN CHECKED.

TST11:

```

2605 012014 TST11: NOP
2606 012014 000240 BIT #BITS+(11*2),TSTNMS ;DO THIS TEST?
2607 012016 033737 001446 001234 BNE 645 ;YES--BRANCH
2608 012024 001002 JMP TST12 ;NO--GO TO THE NEXT TEST
2609 012026 000137 012234 645: MOV #11,#TSTNM ;SET UP TEST NUMBER AND
2610 012032 012737 000011 001102 ;CLEAR THE ERROR FLAG (SERFLG)
2611 JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
2612 012040 004737 024560 MOV #TST11,#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
2613 012044 012737 012154 001110 MOV $TSTNM,#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2614 012052 013777 001102 167062 MOV #RPT,$TIMES ;GET THE ITERATION COUNT
2615 012060 013737 001506 001204 MOV #25,$ERMAX ;MAX ERRORS ALLOWED FOR TEST
2616 012066 112737 000031 001115 MOV #15,$LPADR ;SETUP THE LOOP ADDRESS
2617 012074 012737 012102 001106 15: MOVB FS,DPB.B+10 ;SECTOR ADDRESS
2618 012102 113737 001524 004134 MOVB FS,DPB.C+10 ;SECTOR ADDRESS
2619 012110 113737 001524 004154 MOVB FT,DPB.B+11 ;TRACK ADDRESS
2620 012116 113737 001516 004135 MOVB FT,DPB.C+11 ;TRACK ADDRESS
2621 012124 113737 001516 004155 MOV FC,DPB.B+12 ;STARTING CYLINDER ADDRESS
2622 012132 013737 001510 004136 MOV FC,DPB.C+12 ;STARTING CYLINDER ADDRESS
2623 012140 013737 001510 004156 MOV #EXIT11,BYPASS ;TEST ABORT EXIT
2624 012146 012737 012232 001252 TEST11: MOV #STACK,SP ;SETUP THE STACK POINTER
2625 012154 012706 001100 15: JSR RO,#CALL.C ;GO EXECUTE THE COMMAND
2626 012160 JSR RO,#CALL.B ;GO EXECUTE THE COMMAND
2627 012160 004037 025304
2628 012164 004037 025114

```


2638
2639
2640

.SBTTL *** TIMING TESTS ***

:/#:
:/#:
:/#:

*THE TIMING TESTS WILL ENSURE THAT THOSE FUNCTIONS BEING
*TIMED ARE WITHIN THE TOLERANCES SPECIFIED IN THE "RPO4
*ENGINEERING SPECIFICATIONS".
*THE SEEK TIMING WILL BE PERFORMED USING EXPLICIT SEEK
*OPERATIONS. AT THE COMPLETION OF EACH OF THE TIMING
*TESTS THE MINIMUM, MAXIMUM AND AVERAGE TIMES WILL BE
*TYPED.

:/#:
:/#:
:/#:

2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693

:TEST 12 ROTATIONAL SPEED TIMING TEST

:* THIS TEST WILL START A SEARCH TO CYLINDER 0, TRACK 0, SECTOR
:0. AS SOON AS THE INTERRUPT OCCURS, THE GO BIT IS SET AGAIN
:AND THE OPERATION IS TIMED. THIS PROCEDURE IS REPEATED 10
:TIMES THEN THE AVERAGE TIME IS CALCULATED AND CHECKED TO
:ENSURE IT IS WITHIN TOLERANCE:
: 16.67 MS/REV + OR - 2% IF 60HZ
: 16.67 MS/REV + OR - 2.5% IF 50HZ.

```
*****
TST12:
NOP
BIT 0#BITS+(12*2),TSTNMS ;DO THIS TEST?
BNE 64S ;YES--BRANCH
JMP TST13 ;NO--GO TO THE NEXT TEST
MOV #12,0#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV #TST12,0#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV #STSTNM,0#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV #0#APT,$TIMES ;GET THE ITERATION COUNT
MOV #25,$SERMAX ;MAX ERRORS ALLOWED FOR TEST
TST #0#CLKSTA ;KMI1-P CLOCK?
BGT #1S ;YES--START TEST
JMP TST13 ;NO--GO TO NEXT TEST
MOV #1S,$LPADR ;SETUP LOOP ADDRESS
JSR #0,#SRCHOO ;DO A MASSBUS INIT & RECAL
BR #2S ;RETURN HERE IF NO ERROR
JMP EXIT12 ;RETURN HERE IF ERROR
MOV #0#FC,RPDA(R4) ;FC
MOV #0#FS,-(SP) ;FS
MOV #0#FT,1(SP) ;FT
MOV (SP)+,RPDA(R4) ;LOAD FT/FS
*****
```


2694	012372	012737	012770	001206	MOV	#EXIT12,SESCAPE	::ESCAPE TO EXIT12 ON ERROR
2695	012400	005005			CLR	R5	::COUNT UP
2696	012402	012703	002774		MOV	#T7A,R3	::60HZ PARAMETERS
2697	012406	032737	000100	001220	BIT	#S06,#C.SMR	::60 HZ?
2698	012414	001402			BEQ	TEST12	::YES--BRANCH
2699	012416	012703	003004		MOV	#T7B,R3	::NO--50 HZ PARAMETERS
2700	012422	012706	001100		TEST12: MOV	#STACK,SP	::SETUP STACK
2701	012426	012701	000012		MOV	#10,R1	::TIME 10 SEARCHES
2702	012432	004737	027024		JSR	PC,#STRTMR	::INITIALIZE THE TIMERS
2703	012436	012777	012644	166730	MOV	#78,#PKV	::SETUP VECTOR IN CASE OF OVERFLOW
2704	012444	012777	027022	022014	MOV	#D0ATI,#RPEC	::SETUP RPO4/5/6 VECTOR
2705	012452	005077	166724		15: CLR	#PKB	::START COUNTING AT ZERO
2706	012456	012777	000131	166714	MOV	#131,#PKCS	::INT.EN., COUNT UP AT 100KHZ
2707	012464	012714	000131		MOV	#SEARCH,(R4)	::START A SEARCH
2708	012470	000001			WAIT		::WAIT ON INTERRUPT
2709	012472	042777	000101	166700	BIC	#101,#PKCS	::STOP THE CLOCK
2710	012500	032764	040000	000012	BIT	#BIT14,RPDS1(R4)	::ERROR?
2711	012506	001415			BEQ	25	::NO--BRANCH
2712	012510	104412			SAVREG		::SAVE R0-R5
2713	012512	012702	004164		MOV	#DTADPB,R2	::DPB POINTER
2714	012516	004737	042004		JSR	PC,#SVRH11	::SAVE ALL THE RH11/RPO4 REGISTERS
2715	012522	012764	000040	000010	MOV	#BIT05,RPCS2(R4)	::MASSBUS CLEAR
2716	012530	013764	004164	000010	MOV	#DTADPB,RPCS2(R4)	::SELECT DRIVE
2717	012536	104413			RESREG		::RESTORE R0-R5
2718	012540	104017			ERROR	17	
2719	012542	005077	166634		25: CLR	#PKB	::START THE COUNT AT ZERO
2720	012546	012714	000131		MOV	#SEARCH,(R4)	::START A SEARCH
2721	012552	012777	000131	166620	MOV	#131,#PKCS	::START THE CLOCK
2722	012560	000001			WAIT		::WAIT ON INTERRUPT
2723	012562	042777	000101	166610	BIC	#101,#PKCS	::STOP THE CLOCK
2724	012570	032764	040000	000012	BIT	#BIT14,RPDS1(R4)	::IS "ERR=1"?
2725	012576	001415			BEQ	35	::NO--BRANCH
2726	012600	104412			SAVREG		::SAVE R0-R5
2727	012602	012702	004164		MOV	#DTADPB,R2	::DPB POINTER
2728	012606	004737	042004		JSR	PC,#SVRH11	::SAVE ALL THE RH11/RPO4 REGISTERS
2729	012612	012764	000040	000010	MOV	#BIT05,RPCS2(R4)	::MASSBUS CLEAR
2730	012620	013764	004164	000010	MOV	#DTADPB,RPCS2(R4)	::SELECT DRIVE
2731	012626	104413			RESREG		::RESTORE R0-R5
2732	012630	104017			ERROR	17	::DISK ERROR OCCURRED
2733	012632	004737	027070		35: JSR	PC,#COUNT	::UPDATE THE COUNT
2734	012636	005301			DEC	R1	::DONE?
2735	012640	003304			BGT	15	::NO--BRANCH
2736	012642	000424			BR	85	::YES--GO TO THE EXIT
2737	012644	042777	000101	166526	75: BIC	#101,#PKCS	::STOP THE CLOCK
2738	012652	005037	177776		CLR	#PS	::DROP THE PRIORITY
2739	012656	012600			MOV	(SP)+,R0	::PC OF WAIT+2
2740	012660	005726			TST	(SP)+	::POP THE PS FROM THE STACK
2741	012662	104412			SAVREG		::SAVE R0-R5
2742	012664	012702	004164		MOV	#DTADPB,R2	::DPB POINTER
2743	012670	004737	042004		JSR	PC,#SVRH11	::SAVE ALL THE RH11/RPO4 REGISTERS
2744	012674	012764	000040	000010	MOV	#BIT05,RPCS2(R4)	::MASSBUS CLEAR
2745	012702	013764	004164	000010	MOV	#DTADPB,RPCS2(R4)	::SELECT DRIVE
2746	012710	104413			RESREG		::RESTORE R0-R5
2747	012712	104020			ERROR	20	::CLOCK OVERFLOWED
2748	012714				85:		
2749	012714	012764	000040	000010	MOV	#BIT05,RPCS2(R4)	::MASSBUS INIT.


```

2750 012722 013764 004164 000010      MOV      2#DTADPB,RPCS2(R4) ;SELECT DRIVE
2751 012730 004737 024044              JSR      PC,2#ST.CLK      ;INITIALIZE THE CLOCK
2752 012734 012777 037346 021524      MOV      2#ISR,2#PVEC    ;RESTORE RH11/RPO4/5/6 INT. VECTOR
2753 012742 032737 000100 001220      BIT      2#SMB,2#C.SMR  ;60 HZ?
2754 012750 001004              BNE      9S              ;NO -- BRANCH
2755 012752 004037 027222      JSR      RD,2#TYPTIM     ;GO TYPE THE TIMES
2756 012756 002774              T7A              ;POINTER
2757 012760 004003              BR       EXIT12        ;GO TO EXIT
2758 012762              9S:
2759 012762 004037 027222      JSR      RD,2#TYPTIM     ;GO TYPE THE TIMES
2760 012766 003004              T7B              ;POINTER
2761 012770 000004      EXIT12: SCOPE          ;LOOP ?
2762
2763
2764 ::*****
2765 ;#TEST 13      ONE CYLINDER SEEK TIMING TEST
2766
2767 ;#      THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
2768 ;#      CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
2769 ;#      CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'. THE
2770 ;#      TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT
2771 ;#      EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK.
2772 ;#      THE TIME MUST BE LESS THAN 10MS.
2773
2774 ::*****
2775 ;#TEST13:
2776 012772 000240      NOP
2777 012774 033737 001452 001234      BIT      2#BITS+(13*2),TSTNMS ;DO THIS TEST?
2778 013002 001002      BNE      64S           ;YES--BRANCH
2779 013004 000137 013436              JMP      TST14          ;NO--GO TO THE NEXT TEST
2780 013010 012737 000013 001102 64S:      MOV      2#13,2#TSTNM   ;SET UP TEST NUMBER AND
2781                                ;CLEAR THE ERROR FLAG (SERFLG)
2782 013016 004737 024560      JSR      PC,LOOPRM      ;LOAD THE PARAMETERS FOR THE TEST
2783 013022 012737 012772 001110      MOV      2#TST13,2#SLPERR ;SETUP THE ERROR LOOP ADDRESS
2784 013030 013777 001102 166104      MOV      2#TSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2785 013036 013737 001506 001204      MOV      2#RPT,$TIMES   ;GET THE ITERATION COUNT
2786 013044 112737 000031 001115      MOV      2#25,2#ERMAX   ;MAX ERRORS ALLOWED FOR TEST
2787 013052 005737 001244      TST      2#CLKSTA       ;KH11-P CLOCK?
2788 013056 003002      BGT      1S           ;YES--START TEST
2789 013060 000137 013436              JMP      TST14          ;NO--GO TO NEXT TEST
2790 013064 012737 013064 001106 1S:      MOV      2#1S,2#SLPADR  ;SETUP THE LOOP ADDRESS
2791 013072 004037 026640      JSR      RD,2#SRCHDD    ;DO A MASSBUS INIT. AND RECAL
2792 013076 000402      BR       2S           ;NO ERROR RETURN
2793 013100 000137 013434              JMP      EXIT13        ;ERROR RETURN--SCOPE LOOP CALL
2794 013104 012703 003014 2S:      MOV      2#T10,R3      ;PARAMETER POINTER
2795 013110 012737 013434 001206      MOV      2#EXIT13,2#ESCAPE ;ESCAPE TO EXIT13 ON ERROR
2796 013116 012706 001100      TEST13: MOV      2#STACK,$P     ;SETUP STACK
2797 013122 013737 001510 004176      MOV      FC,2#DTADPB+12 ;START WITH BEGINNING CYLINDER
2798 013130 005237 004176      INC      DTADPB+12     ;INCREMENT THE BEGINNING CYLINDER
2799 013134 005005      CLR      R5           ;SET THE UP/DOWN SWITCH TO UP
2800 013136 004737 027024      JSR      PC,2#STRTHR    ;INITIALIZE THE TIMERS
2801 013142 012777 013330 166224      MOV      2#7S,2#PKV    ;SETUP INCASE OF OVERFLOW
2802 013150 012777 027022 021310      MOV      2#DORTI,2#PVEC ;SET RPO4/5/6 VECTOR
2803 013156 005077 166220 1S:      CLR      2#PKB         ;START THE COUNTER AT ZERO
2804 013162 013764 004176 000034      MOV      2#DTADPB+12,RPC(R4) ;LOAD DESIRED CYLINDER
2805 013170 012714 000105      MOV      2#SEEK,(R4)   ;START A SEEK

```


E05

```

2806 013174 012777 000131 166176 MOV #131,2PKCS ;START THE CLOCK
2807 013202 000001 WAIT ;WAIT ON INTERRUPT
2808 013204 042777 000101 166166 BIC #101,2PKCS ;STOP THE CLOCK
2809 013212 032764 040000 000012 BIT #BIT14,RPDS1(R4) ;ANY DISK ERRORS?
2810 013220 001415 BEQ 25 ;NO--BRANCH
2811 013222 104412 SAVREG ;SAVE R0-R5
2812 013224 012702 004164 MOV #DTADPB,R2 ;DPB POINTER
2813 013230 004737 042004 JSR PC,2#SVRH11 ;SAVE ALL THE RH11/RPO4 REGISTERS
2814 013234 012764 000040 000010 MOV #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2815 013242 013764 004164 000010 MOV #DTADPB,RPCS2(R4) ;SELECT DRIVE
2816 013250 104413 RESREG ;RESTORE R0-R5
2817 013252 104017 ERROR 17 ;REPORT THE ERROR
2818 013254 004737 027070 2S: JSR PC,2#COUNT ;COUNT THIS SEEKS TIME
2819 013260 004737 026416 JSR PC,2#TWOMS ;STALL FOR 2 MILLISECONDS
2820 013264 005705 TST R5 ;UP OR DOWN?
2821 013266 001011 BNE 45 ;DOWN--BRANCH
2822 013270 005237 004176 3S: INC #DTADPB+12 ;MOVE TO NEXT CYLINDER
2823 013274 023737 004176 001512 CMP #DTADPB+12,LC ;OUT OF CYLINDERS?
2824 013302 002725 BLT 15 ;NO--GO DO THE NEXT SEEK
2825 013304 012705 177777 MOV #1,R5 ;SET UP/DOWN SWITCH TO DOWN
2826 013310 000722 BR 15 ;GO DO THE NEXT SEEK
2827 013312 005337 004176 4S: DEC #DTADPB+12 ;MOVE TO NEXT CYLINDER
2828 013316 023727 004176 000000 CMP #DTADPB+12,#0 ;OUT OF CYLINDERS?
2829 013324 003314 BGT 15 ;NO--GO DO THE NEXT SEEK
2830 013326 000424 BR 85 ;GO TO THE EXIT
2831 013330 042777 000101 166042 7S: BIC #101,2PKCS ;STOP THE CLOCK
2832 013336 005037 177776 CLR #PS ;DROP THE PRIORITY
2833 013342 012600 MOV (SP)+,R0 ;PC OF WAIT+2
2834 013344 005726 TST (SP)+ ;POP THE PS FROM THE STACK
2835 013346 104412 SAVREG ;SAVE R0-R5
2836 013350 012702 004164 MOV #DTADPB,R2 ;DPB POINTER
2837 013354 004737 042004 JSR PC,2#SVRH11 ;SAVE ALL THE RH11/RPO4 REGISTERS
2838 013360 012764 000040 000010 MOV #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2839 013366 013764 004164 000010 MOV #DTADPB,RPCS2(R4) ;SELECT DRIVE
2840 013374 104413 RESREG ;RESTORE R0-R5
2841 013376 104020 ERROR 20 ;REPORT CLOCK OVERFLOW
2842 013400 8S:
2843 013400 012764 000040 000010 MOV #BIT05,RPCS2(R4) ;MASSBUS INIT.
2844 013406 013764 004164 000010 MOV #DTADPB,RPCS2(R4) ;SELECT DRIVE
2845 013414 004737 024044 JSR PC,2#ST.CLK ;INITIALIZE THE CLOCK
2846 013420 012777 037346 021040 MOV #ISR,2#PVEC ;RESTORE RH11/RPO4/5/6 INT. VECTOR
2847 013426 004037 027222 JSR R0,2#TYPTIM ;GO TYPE THE TIMES
2848 013432 003014 T10 ;POINTER
2849 013434 000004 EXIT13: SCOPE ;LOOP ?

```

#TEST 14 ACCESS TIME MEASUREMENT TEST

THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
CYLINDER 'LC', THEN A REVERSEK FROM CYLINDER 'LC' TO
CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY
ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASUREMENT.
THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL
OF 256 SEEKS). THE AVERAGE ACCESS TIME MUST BE LESS THAN 30 MS.
CYLINDER 'LC' DEFAULTS TO 136 (10) FOR AN RPO4/5 OR TO 255 (10)

2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861

F05

```

; * FOR AN RPO6.
;*****
TST14:
NOP
BIT 2#BITS+(14*2),TSTNMS ;DO THIS TEST?
BNE 64S ;YES--BRANCH
JMP TST15 ;NO--GO TO THE NEXT TEST
MOV 64S: 814,2#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LOOPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV 8TST14,2#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV STSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV 2#RPT,$TIMES ;GET THE ITERATION COUNT
MOVB 825,$SERMAX ;MAX ERRORS ALLOWED FOR TEST
TST 2#CLKSTA ;KN11-P CLOCK?
BGT 1S ;YES--START TEST
JMP TST15 ;NO--GO TO NEXT TEST
MOV 1S: 81S,SLPADR ;SET THE LOOP ADDRESS
JSR RD,2#SRCHOO ;DO A MASSBUS INIT & RECAL
BR 2S ;RETURN HERE IF NO ERROR
JMP EXIT14 ;RETURN HERE ON ERROR
MOV 2S: 8T11,R3 ;PARAMETER POINTER
MOV 8EXIT14,$ESCAPE ;ESCAPE TO EXIT14 ON ERROR
TEST14: MOV 8STACK,$P ;SETUP STACK
MOV 8128,$R1 ;REPEAT "0-136-0" 128 TIMES
JSR PC,2#STRATMR ;INIT. THE COUNTERS
MOV 87S,$PKV ;SET UP VECTOR IN CASE OF OVERFLOW
MOV 8DORTI,$RVEC ;SETUP RPO4/5/6 VECTOR
1S: CLR 2#PKB ;START COUNT AT ZERO
MOV LC,RPC4(R4) ;'MIDDLE' CYLINDER
MOV 8SEEK,RPCS1(R4) ;START A SEEK
MOV 8131,$PKCS ;START THE CLOCK
WAIT ;WAIT ON INTERRUPT
BIC 8101,$PKCS ;STOP CLOCK
BIT 8BIT14,RPDS1(R4) ;ERR=1?
BEQ 2S ;NO--BRANCH
SAVREG ;SAVE R0-R5
MOV 8DTADPB,R2 ;DPB POINTER
JSR PC,2#SVRH11 ;SAVE ALL THE RH11/RPO4 REGISTERS
MOV 8BIT05,RPCS2(R4) ;MASSBUS CLEAR
MOV 8DTADPB,RPCS2(R4) ;SELECT DRIVE
RESREG ;RESTORE R0-R5
ERROR 17
2S: CLR R5 ;SET UP/DOWN SWITCH TO UP
JSR PC,2#COUNT ;UPDATE THE COUNT
JSR PC,2#TWOMS ;STALL FOR 2 MILLISECONDS
CLR 2#PKB ;START THE COUNT AT ZERO
MOV FC,RPC4(R4) ;BEGINNING CYLINDER
MOV 8SEEK,RPCS1(R4) ;START A SEEK
MOV 8131,$PKCS ;START THE CLOCK
WAIT ;WAIT ON INTERRUPT
BIC 8101,$PKCS ;STOP THE CLOCK
BIT 8BIT14,RPDS1(R4) ;ERR=1?
BEQ 3S ;NO--BRANCH
SAVREG ;SAVE R0-R5
    
```



```

2918 013774 012702 004164      MOV      #DTADPB,R2      ;DPB POINTER
2919 014000 004737 042004      JSR      PC,#SVRH11     ;SAVE ALL THE RH11/RPO4 REGISTERS
2920 014004 012764 000040 000010  MOV      #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2921 014012 013764 004164 000010  MOV      #DTADPB,RPCS2(R4) ;SELECT DRIVE
2922 014020 104413      RESREG      ;RESTORE R0-R5
2923 014022 104017      ERROR      17
2924 014024 012705 177777      35:     MOV      #-1,R5      ;SET UP/DOWN SWITCH TO DOWN
2925 014030 004737 027070      JSR      PC,#COUNT    ;UPDATE THE COUNT
2926 014034 004737 026416      JSR      PC,#TWOMS     ;STALL FOR 2 MILLISECONDS
2927 014040 005301      DEC      R1      ;DONE?
2928 014042 003263      BGT      18      ;NO--BRANCH
2929 014044 000424      BR      B5      ;YES--EXIT
2930 014046 042777 000101 165324 75:     BIC      #101,#PKCS    ;STOP THE CLOCK
2931 014054 005037 177776      CLR      #PS      ;DROP THE PRIORITY
2932 014060 012600      MOV      (SP)+,R0     ;PC OF WAIT+2
2933 014062 005726      TST      (SP)+      ;POP THE PS FROM THE STACK
2934 014064 104412      SAVREG      ;SAVE R0-R5
2935 014066 012702 004164      MOV      #DTADPB,R2      ;DPB POINTER
2936 014072 004737 042004      JSR      PC,#SVRH11     ;SAVE ALL THE RH11/RPO4 REGISTERS
2937 014076 012764 000040 000010  MOV      #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2938 014104 013764 004164 000010  MOV      #DTADPB,RPCS2(R4) ;SELECT DRIVE
2939 014112 104413      RESREG      ;RESTORE R0-R5
2940 014114 104020      ERROR      20      ;CLOCK OVERFLOWED
2941 014116      85:
2942 014116 012764 000040 000010  MOV      #BIT05,RPCS2(R4) ;MASSBUS INIT.
2943 014124 013764 004164 000010  MOV      #DTADPB,RPCS2(R4) ;SELECT DRIVE
2944 014132 004737 024044      JSR      PC,#ST.CLK    ;INITIALIZE THE CLOCK
2945 014136 012777 037346 020322  MOV      #ISR,#RVEC    ;RESTORE RH11/RPO4/5/6 INT. VECTOR
2946 014144 004037 027222      JSR      R0,#TYPTIM   ;GO TYPE THE TIMES
2947 014150 003024      T11      ;POINTER
2948 014152 000004      EXIT14: SCOPE      ;LOOP ?
2949
2950
2951 ;*****
2952 ;*TEST 15      MAXIMUM SEEK TIMING TEST
2953
2954 ;*      THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
2955 ;*      CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
2956 ;*      CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE
2957 ;*      THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK
2958 ;*      TIME. THIS SEQUENCE IS REPEATED 128 TIMES (FOR
2959 ;*      A TOTAL OF 256 SEEKS). THE MAXIMUM SEEK TIME MUST BE LESS THAN
2960 ;*      54 MS. 'LC' DEFAULTS TO 410 (10) FOR RPO4/S'S AND TO 814 (10)
2961 ;*      FOR RPO6'S.
2962 ;*****
2963
2964 TST15:
2965 014154 000240      NOP
2966 014154 033737 001456 001234  BIT      #BITS+(15*2),TSTNMS ;DO THIS TEST?
2967 014164 001002      BNE      645      ;YES--BRANCH
2968 014166 000137 014672 001102 645:   JMP      TST16     ;NO--GO TO THE NEXT TEST
2969 014172 012737 000015 001102  MOV      #15,#STSTNM  ;SET UP TEST NUMBER AND
2970 014200 004737 024560      JSR      PC,LODPRM    ;CLEAR THE ERROR FLAG (SERFLG)
2971 014204 012737 014154 001110  MOV      #TST15,#SLPERR ;LOAD THE PARAMETERS FOR THE TEST
2972 014212 013777 001102 164722  MOV      #STSTNM,#DISPLAY ;SETUP THE ERROR LOOP ADDRESS
2973 014220 013737 001506 001204  MOV      #RPT,$TIMES  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999

```


H05

Address	Offset	PC	Next PC	Next PC	Label	Instruction	Comments
2974	014226	112737	000031	001115		MOVB #25, SERMAX	:MAX ERRORS ALLOWED FOR TEST
2975	014234	005737	001244			TST #CLKSTA	:K11-P CLOCK
2976	014240	003002				BGT 1\$:YES--START TEST
2977	014242	000137	014672			JMP TST16	:NO--GO TO NEXT TEST
2978	014246	012737	014246	001106	1\$:	MOV #1\$, SLPADR	:SETUP THE LOOP ADDRESS
2979	014254	004037	026640			JSR R0, #SRCH00	:DO A MASSBUS INIT & RECAL
2980	014260	000402				BR 2\$:RETURN HERE IF NO ERROR
2981	014262	000137	014670			JMP EXIT15	:RETURN HERE ON ERROR
2982	014266	012703	003034		2\$:	MOV #T12, R3	:PARAMETER POINTER
2983	014272	012737	014670	001206		MOV #EXIT15, SESCPE	:ESCAPE TO EXIT15 ON ERROR
2984	014300	012706	001100		TEST15:	MOV #STACK, SP	:SETUP STACK
2985	014304	012701	000200			MOV #128, R1	:REPEAT "0-'LC'-0" 128 TIMES
2986	014310	004737	027024			JSR PC, #STRTHR	:INIT. THE TIMERS
2987	014314	012777	014564	165052		MOV #7\$, #PKV	:SETUP VECTOR IN CASE OF OVERFLOW
2988	014322	012777	027022	020136		MOV #DORTI, #RVEC	:SETUP RPO4/5/6 VECTOR
2989	014330	005077	165046		1\$:	CLR #PKB	:START COUNTING FROM ZERO
2990	014334	013764	001512	000034		MOV LC, R4(R4)	:MAXIMUM CYLINDER
2991	014342	012764	000105	000000		MOV #SEEK, R4(R4)	:START A SEEK
2992	014350	012777	000131	165022		MOV #131, #PKCS	:START THE CLOCK
2993	014356	000001				WAIT	:WAIT ON INTERRUPT
2994	014360	042777	000101	165012		BIC #101, #PKCS	:STOP THE CLOCK
2995	014366	032764	040000	000012		BIT #BIT14, R4(R4)	:ERR=1?
2996	014374	001415				BEG 2\$:NO--BRANCH
2997	014376	104412				SAVREG	:SAVE R0-R5
2998	014400	012702	004164			MOV #DTADPB, R2	:DPB POINTER
2999	014404	004737	042004			JSR PC, #SVRH11	:SAVE ALL THE RH11/RPO4 REGISTERS
3000	014410	012764	000040	000010		MOV #BIT05, R4(R4)	:MASSBUS CLEAR
3001	014416	013764	004164	000010		MOV #DTADPB, R4(R4)	:SELECT DRIVE
3002	014424	104413				RESREG	:RESTORE R0-R5
3003	014426	104017				ERROR 17	
3004	014430	005005			2\$:	CLR R5	:SET THE UP/DOWN SWITCH TO UP
3005	014432	004737	027070			JSR PC, #COUNT	:UP THE COUNT
3006	014436	004737	026416			JSR PC, #TWOMS	:STALL FOR 2 MILLISECONDS
3007	014442	005077	164734			CLR #PKB	:START COUNT AT ZERO
3008	014446	013764	001510	000034		MOV FC, R4(R4)	:BEGINNING CYLINDER
3009	014454	012764	000105	000000		MOV #SEEK, R4(R4)	:START A SEEK
3010	014462	012777	000131	164710		MOV #131, #PKCS	:START THE CLOCK
3011	014470	000001				WAIT	:WAIT ON INTERRUPT
3012	014472	042777	000101	164700		BIC #101, #PKCS	:STOP THE CLOCK
3013	014500	032764	040000	000012		BIT #BIT14, R4(R4)	:ERR=1?
3014	014506	001415				BEG 3\$:NO--BRANCH
3015	014510	104412				SAVREG	:SAVE R0-R5
3016	014512	012702	004164			MOV #DTADPB, R2	:DPB POINTER
3017	014516	004737	042004			JSR PC, #SVRH11	:SAVE ALL THE RH11/RPO4 REGISTERS
3018	014522	012764	000040	000010		MOV #BIT05, R4(R4)	:MASSBUS CLEAR
3019	014530	013764	004164	000010		MOV #DTADPB, R4(R4)	:SELECT DRIVE
3020	014536	104413				RESREG	:RESTORE R0-R5
3021	014540	104017				ERROR 17	:REPORT THE ERROR
3022	014542	012705	177777		3\$:	MOV #-1, R5	:SET THE UP/DOWN SWITCH TO DOWN
3023	014546	004737	027070			JSR PC, #COUNT	:UPDATE THE COUNT
3024	014552	004737	026416			JSR PC, #TWOMS	:STALL FOR 2 MILLISECONDS
3025	014556	005301				DEC R1	:DONE?
3026	014560	003263				BGT 1\$:NO--BRANCH
3027	014562	000424				BR 8\$:YES--EXIT
3028	014564	042777	000101	164606	7\$:	BIC #101, #PKCS	:STOP THE CLOCK
3029	014572	005037	177776			CLR #PS	:DROP THE PRIORITY


```

3030 014576 012600      MOV      (SP)+,R0      ;PC OF WAIT+2
3031 014600 005726      TST      (SP)+        ;POP THE PS FROM THE STACK
3032 014602 104412      SAVREG                    ;SAVE R0-R5
3033 014604 012702 004164  MOV      @DTADPB,R2    ;DPB POINTER
3034 014610 004737 042004  JSR      PC,@SVRH11    ;SAVE ALL THE RH11/RPO4 REGISTERS
3035 014614 012764 000040 000010  MOV      @BIT05,RPCS2(R4) ;MASSBUS CLEAR
3036 014622 013764 004164 000010  MOV      @DTADPB,RPCS2(R4) ;SELECT DRIVE
3037 014630 104413      RESREG                    ;RESTORE R0-R5
3038 014632 104020      ERROR      20          ;CLOCK OVERFLOWED
3039 014634      BS:
3040 014634 012764 000040 000010  MOV      @BIT05,RPCS2(R4) ;MASSBUS INIT.
3041 014642 013764 004164 000010  MOV      @DTADPB,RPCS2(R4) ;SELECT DRIVE
3042 014650 004737 024044      JSR      PC,@ST.CLK    ;INITIALIZE THE CLOCK
3043 014654 012777 037346 017604  MOV      @ISR,@RPEC    ;RESTORE RH11/RPO4/5/6 INT. VECTOR
3044 014662 004037 027222      JSR      R0,@TYPTIM    ;GO TYPE THE TIMES
3045 014666 003034      T12
3046 014670 000004      EXIT15: SCOPE        ;LOOP ?

```


3047
3048
3049

.SBTTL *** ADDRESSING TESTS ***

:/#:
:/#:
:/#:

;*THE ADDRESSING TESTS ENSURES PROPER OPERATION OF THE TRACK
;*AND SECTOR ADDRESS CIRCUITRY. BOTH ADDRESSING TESTS
;*WILL BE PERFORMED ON CYLINDER FC.

:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:
:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:
:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:/#\#:

3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102

014672
014672 000240
014674 033737 001460 001234
014702 001002
014704 000137 015270
014710 012737 000016 001102
014716 004737 024560
014722 012737 014766 001110
014730 013777 001102 164204
014736 013737 001506 001204
014744 113737 001364 001115
014752 012737 015266 001252
014760 012737 014766 001106
014766 012706 001100
014772 004737 027654
014776 013737 001510 004176
015004 113737 001516 004175
015012 105037 004174
015016 013737 001352 004170
015024 012737 047662 004172
015032 012737 015032 001110
015040 012706 001100
015044 012737 000161 004166
015052 004037 025474
015056 012737 000151 004166

*TEST 16 SECTOR ADDRESSING TEST

THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT". THE DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR BEING WRITTEN. A WRITE CHECK IS PERFORMED, THE BUFFER IS CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED. THEN SECTOR 0 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED. THEN SECTOR 1 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED. THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH REWRITE SECTOR 21 AND WRITE CHECK SECTORS 0-21.

TST16:

```
NOP  
BIT 2#BITS<(16*2),TSTNMS ;DO THIS TEST?  
BNE 64S ;YES--BRANCH  
JMP TST17 ;NO--GO TO THE NEXT TEST  
MOV 64S: 016,2#TSTNM ;SET UP TEST NUMBER AND  
;CLEAR THE ERROR FLAG (SERFLG)  
JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST  
MOV #TST16,2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS  
MOV $TSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER  
MOV 2#NPT,$TIMES ;GET THE ITERATION COUNT  
MOVB ERR.CT,SERMAX ;MAX ERRORS ALLOWED FOR TEST  
MOV #EXIT16,2#BYPASS  
MOV #TST16,SLPADR ;SETUP THE LOOP ADDRESS  
TEST16: MOV #STACK,SP ;SET THE STACK POINTER  
JSR PC,2#FILBUF ;FILL THE BUFFER WITH SECTOR ADDRESSES  
MOV 2#FC,2#DTADPB+12 ;CYLINDER  
MOVB 2#FT,2#DTADPB+11 ;TRACK  
CLRB 2#DTADPB+10 ;SECTOR  
MOV TRCKMC,2#DTADPB+4 ;WORD COUNT  
MOV #BUFFER,2#DTADPB+6 ;BUFFER ADDRESS  
MOV 0,SLPERR ;SETUP THE ERROR LOOP ADDRESS  
MOV #STACK,SP ;LOAD THE STACK POINTER  
MOV #WRITE,2#DTADPB+2 ;COMMAND=WRITE DATA  
JSR RD,2#DRVCAL ;START A DATA TRANSFER  
MOV #WRCKD,2#DTADPB+2 ;COMMAND=WRITE CHECK DATA
```



```

3103 015064 012737 015064 001110 MOV #,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3104 015072 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
3105 015076 004037 025474 JSR RD,#DRVCAL ;START A DATA TRANSFER
3106 015102 012737 015102 001110 MOV #,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3107 015110 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
3108 015114 004037 027712 JSR RD,#CLBUF ;CLEAR BUFFER
3109 015120 012737 000171 004166 MOV #READ,#DTADPB+2 ;COMMAND = READ
3110 015126 004037 025474 JSR RD,#DRVCAL ;START A DATA TRANSFER
3111 015132 004037 027760 JSR RD,#CKSCTR ;CHECK THE SECTOR DATA READ
3112 015136 012700 047662 MOV #BUFFER,R0 ;BUFFER ADDRESS
3113 015142 005001 CLR R1 ;FIRST SECTOR
3114 015144 012737 015144 001110 MOV #,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3115 015152 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
3116 015156 012737 000161 004166 IS: MOV #WRITE,#DTADPB+2 ;COMMAND=WRITE DATA
3117 015164 012737 177400 004170 MOV #SCTRMC,#DTADPB+4 ;WORD COUNT
3118 015172 010037 004172 MOV RD,#DTADPB+6 ;BUFFER ADDRESS
3119 015176 110137 004174 MOVVB R1,#DTADPB+10 ;SECTOR
3120 015202 004037 025474 JSR RD,#DRVCAL ;START A DATA TRANSFER
3121 015206 012737 015206 001110 MOV #,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3122 015214 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
3123 015220 012737 000151 004166 MOV #WRCKD,#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3124 015226 013737 001352 004170 MOV TRCKMC,#DTADPB+4 ;WORD COUNT
3125 015234 012737 047662 004172 MOV #BUFFER,#DTADPB+6 ;BUFFER ADDRESS
3126 015242 105037 004174 CLRB #DTADPB+10 ;SECTOR
3127 015246 004037 025474 JSR RD,#DRVCAL ;START A DATA TRANSFER
3128 015252 062700 001000 ADD #512.,R0 ;MOVE TO NEXT SECTOR
3129 015256 005201 INC R1
3130 015260 023701 001630 CMP PRMLMT+22,R1 ;DONE?
3131 015264 103334 BHS IS ;NO--BRANCH
3132 015266 000004 EXIT16: SCOPE ;LOOP ?
3133
3134 ;*****
3135 ;*TEST 17 TRACK ADDRESSING TEST
3136
3137 ;* THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
3138 ;* IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
3139 ;* GETTING ITS OWN TRACK ADDRESS.
3140 ;* A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO ENSURE
3141 ;* THE DATA IS VALID. THEN TRACK 0 IS REWRITTEN AND TRACK 1
3142 ;* THROUGH TRACK 18 IS WRITE CHECKED. THEN TRACK 1 IS
3143 ;* REWRITTEN AND TRACK 2 THROUGH TRACK 18 IS WRITE CHECKED.
3144 ;* THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING TRACK 17
3145 ;* AND WRITE CHECKING TRACK 18.
3146
3147 ;*****
3148 TST17:
3149 015270 000240 NOP
3150 015272 033737 001462 001234 BIT #BITS+(17*2),TSTNMS ;DO THIS TEST?
3151 015300 001002 BNE 645 ;YES--BRANCH
3152 015302 000137 015710 JMP TST20 ;NO--GO TO THE NEXT TEST
3153 015306 012737 000017 001102 645: MOV #17,#STSTNM ;SET UP TEST NUMBER AND
3154 ;CLEAR THE ERROR FLAG (SERFLG)
3155 015314 004737 024560 JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
3156 015320 012737 015364 001110 MOV #TST17,#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
3157 015326 013777 001102 163606 MOV $STSTNM,#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3158 015334 013737 001506 001204 MOV #RPT,$TIMES ;GET THE ITERATION COUNT

```


L05

3159	015342	113737	001364	001115		MOV	ERR.CT, SERMAX	;MAX ERRORS ALLOWED FOR TEST
3160	015350	012737	015706	001252		MOV	#EXIT17, 2#BYPASS	
3161	015356	012737	015364	001106		MOV	#TEST17, \$LPADR	; SETUP THE LOOP ADDRESS
3162	015364	012706	001100		TEST17:	MOV	#STACK, SP	; SET THE STACK POINTER
3163	015370	004737	027654			JSR	PC, 2#FILBUF	; FILL THE BUFFER WITH TRACK ADDRESS
3164	015374	012737	000161	004166		MOV	#WRITE, 2#DTADPB+2	; COMMAND=WRITE DATA
3165	015402	013737	001510	004176		MOV	2#FC, 2#DTADPB+12	; CYLINDER
3166	015410	113737	001524	004174		MOV	2#FS, 2#DTADPB+10	; SECTOR
3167	015416	012737	177400	004170		MOV	#SCTRC, 2#DTADPB+4	; WORD COUNT
3168	015424	012737	047662	004172		MOV	#BUFFER, 2#DTADPB+6	; BUFFER ADDRESS
3169	015432	005000				CLR	RO	; TRACK=0
3170	015434	012737	015434	001110		MOV	8, \$LPERR	; SETUP THE ERROR LOOP ADDRESS
3171	015442	012706	001100			MOV	#STACK, SP	; LOAD THE STACK POINTER
3172	015446	110037	004175		15:	MOV	RO, 2#DTADPB+11	; TRACK ADDRESS
3173	015452	004037	025474			JSR	RO, 2#DRVCAL	; START A DATA TRANSFER
3174	015456	062737	001000	004172		ADD	#256.*2., 2#DTADPB+6	; UPDATE BUFFER ADDRESS
3175	015464	005200				INC	RO	; UPDATE TRACK NUMBER
3176	015466	022700	000023			CMP	#19., RO	; OUT OF TRACKS?
3177	015472	003365				BGT	15	; NO--BRANCH
3178	015474	012737	047662	004172		MOV	#BUFFER, 2#DTADPB+6	; BUFFER ADDRESS
3179	015502	005000				CLR	RO	
3180	015504	012737	015504	001110		MOV	8, \$LPERR	; SETUP THE ERROR LOOP ADDRESS
3181	015512	012706	001100			MOV	#STACK, SP	; LOAD THE STACK POINTER
3182	015516	012737	000151	004166		MOV	#WRCKD, 2#DTADPB+2	; COMMAND=WRITE CHECK
3183	015524	110037	004175		25:	MOV	RO, 2#DTADPB+11	; TRACK ADDRESS
3184	015530	004037	025474			JSR	RO, 2#DRVCAL	; START A DATA TRANSFER
3185	015534	062737	001000	004172		ADD	#256.*2., 2#DTADPB+6	; UPDATE BUFFER ADDRESS
3186	015542	005200				INC	RO	; UPDATE TRACK NUMBER
3187	015544	022700	000023			CMP	#19., RO	; OUT OF TRACKS?
3188	015550	003365				BGT	25	; NO--BRANCH
3189	015552	005000				CLR	RO	; FIRST TRACK ADDRESS
3190	015554	110037	004175		35:	MOV	RO, 2#DTADPB+11	; TRACK
3191	015560	010001				MOV	RO, R1	; FORM BUFFER ADDRESS
3192	015562	012737	047662	004172		MOV	#BUFFER, 2#DTADPB+6	; BUFFER ADDRESS
3193	015570	005301			45:	DEC	R1	
3194	015572	002411				BLT	55	
3195	015574	062737	001000	004172		ADD	#256.*2., 2#DTADPB+6	
3196	015602	000772				BR	45	
3197	015604	012737	015604	001110		MOV	8, \$LPERR	; SETUP THE ERROR LOOP ADDRESS
3198	015612	012706	001100			MOV	#STACK, SP	; LOAD THE STACK POINTER
3199	015616	012737	000161	004166	55:	MOV	#WRITE, 2#DTADPB+2	; COMMAND=WRITE DATA
3200	015624	004037	025474			JSR	RO, 2#DRVCAL	; START A DATA TRANSFER
3201	015630	062737	001000	004172	65:	ADD	#256.*2., 2#DTADPB+6	; UPDATE BUFFER ADDRESS
3202	015636	105237	004175			INCB	2#DTADPB+11	; MOVE TO NEXT TRACK
3203	015642	012737	015642	001110		MOV	8, \$LPERR	; SETUP THE ERROR LOOP ADDRESS
3204	015650	012706	001100			MOV	#STACK, SP	; LOAD THE STACK POINTER
3205	015654	012737	000151	004166		MOV	#WRCKD, 2#DTADPB+2	; COMMAND=WRITE CHECK DATA
3206	015662	004037	025474			JSR	RO, 2#DRVCAL	; START A DATA TRANSFER
3207	015666	122737	000022	004175		CMPB	#18., 2#DTADPB+11	; OUT OF TRACKS?
3208	015674	003355				BGT	65	; NO--BRANCH
3209	015676	005200				INC	RO	; NEXT TRACK TO WRITE
3210	015700	022700	000022			CMP	#18., RO	; OUT OF TRACKS?
3211	015704	003323				BGT	35	; NO--BRANCH
3212	015706	000004			EXIT17:	SCOPE		

*** DATA TEST ***

.SBTTL *** DATA TEST ***

:TEST 20 DATA TEST

THIS TEST PERFORMS DATA STORAGE AND RETRIEVAL ON CYLINDERS "FC" THROUGH "LC" BY THE INCREMENT "IC" USING THE DATA PATTERNS SPECIFIED. THE FOLLOWING SEQUENCE OCCURS FOR EACH CYLINDER:
1. SET "NT" TO "FT" THEN REPEAT 2-4 UNTIL "NT" > "LT"
2. WRITE THEN WRITE CHECK "FS" THROUGH "LS" OF TRACK "NT"
3. READ THEN SOFTWARE COMPARE "FS" THROUGH "LS" OF TRACK "NT"
4. INCREMENT "NT" BY "IT"
5. REPEAT STEPS 1-4 FOR EACH DATA PATTERN
6. REPEAT STEPS 1-5 FOR "FC" THROUGH "LC" ADVANCING BY "IC"

IF A WRITE CHECK ERROR OCCURS THE ERROR IS REPORTED AND THE TRACK IN ERROR IS REWRITTEN AND CHECKED. THIS CHECK IS ACCOMPLISHED BY PERFORMING TWO(2) SUCCESSIVE ERROR FREE WRITE CHECKS. IF THE CHECK FAILS THE ERROR IS REPORTED AS FATAL AND NO READ OCCURS.
FS DEFAULTS TO 1 AND LS DEFAULTS TO 0
PAT DEFAULTS TO 17777 (ALL POSSIBLE PATTERNS)
THE POSSIBLE PATTERNS ARE:

PAT 0	PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7
165555	000001	177776	000000	000000	052525	007417	026455
133333	000003	177774	000000	010421	052525	007417	026455
165555	000007	177770	000000	021042	052525	007417	026455
133333	000017	177760	177777	031463	125252	170360	151322
165555	000037	177740	177777	042104	125252	170360	151322
133333	000077	177700	177777	052525	125252	170360	151322
165555	000177	177600	000000	063146	052525	007417	026455
133333	000377	177400	000000	073567	052525	007417	026455
165555	000777	177000	177777	104210	125252	170360	151322
133333	001777	176000	177777	114631	125252	170360	151322
165555	003777	174000	000000	125252	052525	007417	026455
133333	007777	170000	177777	135673	125252	170360	151322
165555	017777	160000	000000	146314	052525	007417	026455
133333	037777	140000	177777	156735	125252	170360	151322
165555	077777	100000	000000	167356	052525	007417	026455
133333	177777	000000	177777	177777	125252	170360	151322
PAT 8	PAT 9	PAT 10	PAT 11	PAT 12	PAT 13	PAT 14	PAT 15
165555	000001	177776	172666	077777	153333	000000	177777
133333	000002	177775	155555	137777	066667	177777	000000
165555	000004	177773	172666	157777	153333	177777	000000
133333	000010	177767	155555	167777	066667	177777	000000
165555	000020	177757	172666	173777	153333	177777	000000
133333	000040	177737	155555	175777	066667	177777	000000
165555	000100	177677	172666	176777	153333	177777	000000
133333	000200	177577	155555	177377	066667	177777	000000
165555	000400	177377	172666	177577	153333	177777	000000
133333	001000	176777	155555	177677	066667	177777	000000
165555	002000	175777	172666	177737	153333	177777	000000

3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268


```

3269      *      133333 004000 173777 155555 177757 066667 177777 000000
3270      *      165555 010000 167777 172666 177767 153333 177777 000000
3271      *      133333 020000 157777 155555 177773 066667 177777 000000
3272      *      165555 040000 137777 172666 177775 153333 177777 000000
3273      *      133333 100000 077777 155555 177776 066667 177777 000000
3274      *
3275      *
3276      *
3277      *

```

```

3277 015710          *ST20:
3278 015710 000240      NOP
3279 015712 033737 001424 001236      BIT      BITS+(20*2-40),TSTNMS+2 ;DO THIS TEST ?
3280 015720 001002      BNE      64$ ;YES--BRANCH
3281 015722 000137 016430      JMP      TST21 ;NO--GO TO THE NEXT TEST
3282 015726 012737 000020 001102 64$: MOV      #20,#$TSTNM ;SET UP TEST NUMBER AND
3283                                     ;CLEAR THE ERROR FLAG (SERFLG)
3284 015734 004737 024560      JSR      PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
3285 015740 012737 016112 001110      MOV      #TST20,#$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
3286 015746 013777 001102 163166      MOV      $TSTNM,$DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3287 015754 013737 001506 001204      MOV      #RPT,$TIMES ;GET THE ITERATION COUNT
3288 015762 113737 001364 001115      MOV      ERR.CT,SERMAX ;MAX ERRORS ALLOWED FOR TEST
3289 015770 012737 015770 001106      MOV      #,$SLPADR ;SETUP THE LOOP ADDRESS
3290 015776 005000      CLR      R0 ;CLEAR SWITCH
3291 016000 005004      CLR      R4 ;FORM WORD COUNT IN R4
3292 016002 013701 001526      MOV      #LS,R1
3293 016006 163701 001524      SUB      #FS,R1
3294 016012 002004      BGE      1$ ;BRANCH IF FS < OR = LS
3295 016014 063701 001630      ADD      PRMLT+22,R1 ;ADD MAXIMUM SECTOR ADDRESS TO
3296 016020 005201      INC      R1 ;MAKE THE DIFFERENCE POSITIVE
3297 016022 005100      COM      R0 ;SET SWITCH
3298 016024 062704 000400      1$: ADD      #256.,R4
3299 016030 005301      DEC      R1
3300 016032 002374      BGE      1$
3301 016034 005404      NEG      R4
3302 016036 010405      MOV      R4,R5 ;COPY NORMAL WORD COUNT INTO SMALL WC
3303 016040 005700      TST      R0 ;SWITCH SET?
3304 016042 001412      BEQ      3$ ;NO--BRANCH
3305 016044 005005      CLR      R5 ;FORM WORD COUNT FOR LS < FS
3306 016046 013701 001630      MOV      PRMLT+22,R1
3307 016052 163701 001524      SUB      #FS,R1
3308 016056 062705 000400      2$: ADD      #256.,R5
3309 016062 005301      DEC      R1
3310 016064 002374      BGE      2$
3311 016066 005405      NEG      R5
3312 016070 113737 001524 004174 3$: MOV      #FS,#DTADPB+10 ;SECTOR
3313 016076 012737 047662 004172      MOV      #BUFFER,#DTADPB+6 ;DATA BUFFER
3314 016104 012737 016426 001252      MOV      #EXIT20,#BYPASS
3315 016112 012706 001100      TEST20: MOV      #STACK,$P ;LOAD THE STACK POINTER
3316 016116 005037 001334      CLR      #WCEFLG ;CLEAR THE WRITE CHECK ERROR FLAG
3317 016122 013701 001510      MOV      #FC,R1 ;PICKUP FIRST CYLINDER.
3318 016126 000407      BR      2$
3319 016130 005720      1$: TST      (R0)+ ;MOVE TO NEXT DATA PATTERN
3320 016132 022700 000040      CMP      #16.*2.,R0 ;OUT OF PATTERNS?
3321 016136 003004      BGT      3$ ;NO--BRANCH
3322 016140 004037 027600      JSR      R0,#INCCYL ;MOVE TO NEXT CYLINDER
3323 016144 000530      BR      EXIT20 ;OUT OF CYLINDERS
3324 016146 005000      2$: CLR      R0 ;START WITH PATTERN 0

```


000

```

3325 016150 036037 001424 001530 3$: BIT BITS(R0),2#PAT ;THIS PATTERN SELECTED?
3326 016156 001764 BEQ 1$ ;NO--BRANCH
3327 016160 013702 MOV 2#FT,R2 ;FIRST TRACK
3328 016164 010137 004176 MOV R1,2#DTADPB+12 ;CYLINDER
3329 016170 110237 004175 4$: MOV R2,2#DTADPB+11 ;TRACK
3330 016174 010437 004170 MOV R4,2#DTADPB+4 ;WORD COUNT
3331 016200 023701 001512 CMP LC,R1 ;LAST DISK CYLINDER?
3332 016204 003005 BGT 5$ ;NO--BRANCH
3333 016206 022702 000022 CMP 818.,R2 ;LAST DISK TRACK?
3334 016212 003002 BGT 5$ ;NO--BRANCH
3335 016214 010537 004170 MOV R5,2#DTADPB+4 ;SHORT WORD COUNT
3336 016220 017703 162714 5$: MOV 2#SW,R3 ;INHIBIT WRITE AND
3337 016224 005103 COM R3 ;WRITE CHECK?
3338 016226 032703 000030 BIT 2#SW04!SW03,R3
3339 016232 001436 BEQ 7$ ;YES--BRANCH
3340 016234 004737 030330 JSR PC,2#SETBUF ;MOVE DATA PATTERN INTO THE BUFFER
3341 016240 032777 000020 162672 BIT 2#SW04,2#SWR ;INHIBIT WRITE?
3342 016246 001012 BNE 6$ ;YES--BRANCH
3343 016250 012737 016250 001110 MOV 8.,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3344 016256 012706 001100 MOV 2#STACK,SP ;LOAD THE STACK POINTER
3345 016262 012737 000161 004166 MOV 2#WRITE,2#DTADPB+2 ;COMMAND=WRITE DATA
3346 016270 004037 025474 JSR R0,2#DRVCAL ;START A DATA TRANSFER
3347 016274 032777 000010 162636 6$: BIT 2#SW03,2#SWR ;INHIBIT WRITE CHECK?
3348 016302 001012 BNE 7$ ;YES--BRANCH
3349 016304 012737 016304 001110 MOV 8.,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3350 016312 012706 001100 MOV 2#STACK,SP ;LOAD THE STACK POINTER
3351 016316 012737 000151 004166 MOV 2#WRCKD,2#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3352 016324 004037 025474 JSR R0,2#DRVCAL ;START A DATA TRANSFER
3353 016330 005737 001334 7$: TST 2#WCEFLG ;WRITE CHECK ERROR FLAG SET?
3354 016334 001404 BEQ 8$ ;NO--BRANCH
3355 016336 032777 000001 162574 BIT 2#SW00,2#SWR ;PERFORM READ AFTER FATAL "WCE"?
3356 016344 001424 BEQ 9$ ;NO--BRANCH
3357 016346 032777 000004 162564 8$: BIT 2#SW02,2#SWR ;INHIBIT READ DATA AND SOFTWARE COMPARE?
3358 016354 001020 BNE 9$ ;YES--BRANCH
3359 016356 012737 016356 001110 MOV 8.,SLPERR ;SETUP THE ERROR LOOP ADDRESS
3360 016364 012706 001100 MOV 2#STACK,SP ;LOAD THE STACK POINTER
3361 016370 012737 000171 004166 MOV 2#READ,2#DTADPB+2 ;COMMAND=READ
3362 016376 004037 025474 JSR R0,2#DRVCAL ;START A DATA TRANSFER
3363 016402 032777 000002 162530 BIT 2#SW01,2#SWR ;COMPARE THE DATA?
3364 016410 001002 BNE 9$ ;NO--BRANCH
3365 016412 004737 030420 JSR PC,2#DATCMP ;YES--DO IT
3366 016416 004037 027550 9$: JSR R0,2#INCTRK ;MOVE TO NEXT TRACK
3367 016422 000642 BR 1$ ;OUT OF TRACKS GO TO NEXT PATTERN
3368 016424 000661 BR 4$ ;LOOP
3369 016426 000004 EXIT20: SCOPE ;SCOPE LOOP
    
```


.SBTTL *** EXERCISE TEST ***

:TEST 21 RANDOM ADDRESS AND RANDOM PATTERN TEST

STARTING AT "FC" AND GOING THROUGH "LC" THE DISK PACK IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR FOR THAT SECTOR. THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES "R" DEFAULTS TO 20,000.

- 1) GENERATE A RANDOM ADDRESS
- 2) WRITE A RANDOM PATTERN AT THE ADDRESS GENERATED IN 1.
- 3) GENERATE A RANDOM ADDRESS
- 4) READ THE SECTOR AT THE ADDRESS GENERATED IN 3.
- 5) DO A SOFTWARE CHECK OF THE DATA READ IN 4.
- 6) DO A WRITE CHECK OF THE DATA WRITTEN IN 2
- 7) GENERATE A RANDOM ADDRESS
- 8) READ THE SECTOR AT THE ADDRESS GENERATED IN 7.
- 9) DO A SOFTWARE CHECK OF THE DATA READ IN 8
- 10) DO A WRITE CHECK OF THE DATA WRITTEN IN 2

TST21:

3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425

016430				
016430	000240			
016432	033737	001426	001236	
016440	001002			
016442	000137	017206		
016446	012737	000021	001102	645:
016454	004737	024560		
016460	012737	016700	001110	
016466	013777	001102	162446	
016474	013737	001506	001204	
016502	113737	001364	001115	
016510	012737	016510	001106	
016516	012737	017204	001252	
016524	012737	176543	023556	
016532	012737	123456	023560	
016540	013737	001510	004176	
016546	013737	001352	004170	
016554	012737	047662	004172	
016562	012737	000161	004166	
016570	032737	100000	001220	
016576	001027			
016600	004037	030736		
016604	005037	004174		15:
016610	012737	016610	001110	
016616	012706	001100		
016622				25:
016622	004037	025474		
016626	105237	004175		

```

NOP
BIT BITS+(21*2-40),TSTNMS+2 ;DO THIS TEST ?
BNE 645 ;YES--BRANCH
JMP TST22 ;NO--GO TO THE NEXT TEST
MOV 645,2#STSTNM ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG (SERFLG)
JSR PC,LOOPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV #TST21,2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV STSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV #RPT,STINES ;GET THE ITERATION COUNT
MOVB ERR,CT,SERMAX ;MAX ERRORS ALLOWED FOR TEST
MOV #,SLPADR ;SETUP THE LOOP ADDRESS
MOV #EXIT21,2#BYPASS
MOV #176543,2#SHINUM ;PRIME THE RANDOM NUMBER GENERATOR
MOV #123456,2#SLONUM
MOV #FC,2#DTADPB+12 ;CYLINDER
MOV TRACK,2#DTADPB+4 ;WORD COUNT
MOV #BUFFER,2#DTADPB+6 ;BUFFER ADDRESS
MOV #WRITE,2#DTADPB+2 ;COMMAND
BIT #SM15,2#C.SWR ;WRITE THE DISK PACK BEFORE TESTING?
BNE 35 ;NO--BRANCH
JSR RO,2#FILRAN ;FILL DATA BUFFER WITH RANDOM DATA
CLR 2#DTADPB+10 ;SECTOR AND TRACK
MOV #,SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV #STACK,SP ;LOAD THE STACK POINTER
JSR RO,2#DRYCAL ;START A DATA TRANSFER
INCB 2#DTADPB+11 ;NEXT TRACK

```



```

3426 016632 122737 000023 004175  CMPB  @19.,@DTADPB+11 ;TIME FOR NEXT CYLINDER
3427 016640 003370          BGT    25 ;NO--BRANCH
3428 016642 005237 004176  INC    @DTADPB+12
3429 016646 023737 001512 004176  CMP    @@LC,@DTADPB+12 ;OUT OF CYLINDERS?
3430 016654 002353          BGE    18 ;NO--BRANCH
3431 016656 012737 177400 004170 3S:  MOV    @SCTRC,@DTADPB+4 ;WORD COUNT
3432 016664 012737 016700 001106  MOV    @TEST21,@SLPADR
3433 016672 012737 016700 001110  MOV    @TEST21,@SLPERR
3434 016700 012706 001100  TEST21: MOV   @STACK,SP ;SET STACK POINTER
3435 016704 004037 031212  JSR    RD,@RANADR ;GENERATE A RANDOM ADDRESS
3436 016710 013737 004174 001340  MOV    @DTADPB+10,@SVADR ;SAVE THE TRACK/SECTOR
3437 016716 013737 004176 001342  MOV    @DTADPB+12,@SVADR+2 ;SAVE THE CYLINDER
3438 016724 012737 000161 004166  MOV    @WRITE,@DTADPB+2 ;COMMAND=WRITE DATA
3439 016732 012701 047662  MOV    @BUFFER,R1 ;BUFFER ADDRESS
3440 016736 010137 004172  MOV    R1,@DTADPB+6
3441 016742 004037 031156  JSR    RD,@RANPAT ;GENERATE RANDOM PATTERN
3442 016746 012737 016746 001110  MOV    @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3443 016754 012706 001100  MOV    @STACK,SP ;LOAD THE STACK POINTER
3444 016760 004037 025474  JSR    RD,@DRVCAL ;START A DATA TRANSFER
3445 016764 004037 031212  JSR    RD,@RANADR
3446 016770 012737 000171 004166  MOV    @READ,@DTADPB+2 ;COMMAND=READ DATA
3447 016776 012737 050662 004172  MOV    @BUFFER+512.,@DTADPB+6 ;BUFFER ADDRESS
3448 017004 012737 017004 001110  MOV    @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3449 017012 012706 001100  MOV    @STACK,SP ;LOAD THE STACK POINTER
3450 017016 004037 025474  JSR    RD,@DRVCAL ;START A DATA TRANSFER
3451 017022 004037 030760  JSR    RD,@RANCK ;CHECK THE DATA
3452 017026 013737 001340 004174  MOV    @SVADR,@DTADPB+10 ;GET ADDRESS OF WHERE THE LAST
3453 017034 013737 001342 004176  MOV    @SVADR+2,@DTADPB+12 ;WRITE WAS PERFORMED
3454 017042 012737 000151 004166  MOV    @WRCKD,@DTADPB+2 ;COMMAND=WRITE CHECK DATA
3455 017050 012737 047662 004172  MOV    @BUFFER,@DTADPB+6 ;DATA BUFFER ADDRESS
3456 017056 012737 017056 001110  MOV    @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3457 017064 012706 001100  MOV    @STACK,SP ;LOAD THE STACK POINTER
3458 017070 004037 025474  JSR    RD,@DRVCAL ;START A DATA TRANSFER
3459 017074 004037 031212  JSR    RD,@RANADR ;GENERATE A RANDOM ADDRESS
3460 017100 012737 000171 004166  MOV    @READ,@DTADPB+2 ;COMMAND=READ
3461 017106 012737 050662 004172  MOV    @BUFFER+512.,@DTADPB+6 ;DATA BUFFER ADDRESS
3462 017114 012737 017114 001110  MOV    @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3463 017122 012706 001100  MOV    @STACK,SP ;LOAD THE STACK POINTER
3464 017126 004037 025474  JSR    RD,@DRVCAL ;START A DATA TRANSFER
3465 017132 004037 030760  JSR    RD,@RANCK ;CHECK THE DATA
3466 017136 013737 001340 004174  MOV    @SVADR,@DTADPB+10 ;GET DISK ADDRESS OF THE
3467 017144 013737 001342 004176  MOV    @SVADR+2,@DTADPB+12 ;LAST WRITE
3468 017152 012737 000151 004166  MOV    @WRCKD,@DTADPB+2 ;COMMAND=WRITE CHECK DATA
3469 017160 012737 047662 004172  MOV    @BUFFER,@DTADPB+6 ;DATA BUFFER ADDRESS
3470 017166 012737 017166 001110  MOV    @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3471 017174 012706 001100  MOV    @STACK,SP ;LOAD THE STACK POINTER
3472 017200 004037 025474  JSR    RD,@DRVCAL ;START A DATA TRANSFER
3473 017204 000004  EXIT21: SCOPE ;LOOP ?

```

.SBTTL *** RPO4 ACCESS TIME ADJUSTMENT TEST ***

;;*****
;#TEST 22 RPO4 ACCESS TIME ADJUSTMENT TEST

;; THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 136 TO ALLOW THE
;# OPERATOR TO ADJUST THE ACCESS TIME ON AN RPO4 USING THE

E06

```

3482          ;*      DDU.  THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
3483          ;*      SO THAT THE ACCESS TIME INDICATORS ON THE DDU MAY BE OBSERVED.
3484
3485          ::*****
3486          TST22:
3487          NOP
3488          BIT     BITS+(22*2-40), TSTNMS+2 ;DO THIS TEST ?
3489          BNE     645 ;YES--BRANCH
3490          JMP     SEOP ;NO--GO TO THE END OF THE PROGRAM
3491          MOV     645: #22, 2#STSTNM ;SET UP TEST NUMBER AND
3492          ;CLEAR THE ERROR FLAG (SERFLG)
3493          JSR     PC.LODPRM ;LOAD THE PARAMETERS FOR THE TEST
3494          MOV     #TEST22, 2#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
3495          MOV     STSTNM, 2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3496          MOV     2#RPT, $TIMES ;GET THE ITERATION COUNT
3497          MOV     #100, 2#SERMAX ;MAX ERRORS ALLOWED FOR TEST
3498          MOV     #TEST22, 2#SLPADR ;SETUP THE LOOP ADDRESS
3499          TEST22: MOV     #STACK, SP ;SETUP THE STACK POINTER
3500          MOV     LC, DPB.A+12 ;ENDING CYLINDER
3501          MOV     #SEEK, 2#DPB.A+2 ;SEEK=COMMAND
3502          JSR     RD, 2#CALL.A ;GO EXECUTE THE COMMAND
3503          JSR     RD, STALL ;STALL
3504          .WORD  STALL3 ;ADDRESS OF STALL VALUE
3505          MOV     FC, DPB.A+12 ;STARTING CYLINDER
3506          MOV     #SEEK, 2#DPB.A+2 ;SEEK=COMMAND
3507          JSR     RD, 2#CALL.A ;GO EXECUTE THE COMMAND
3508          JSR     RD, STALL ;STALL
3509          .WORD  STALL3 ;ADDRESS OF STALL VALUE
3510          EXIT22: SCOPE ;LOOP ?
3511
3512          .SBTTL  END OF PASS ROUTINE
3513
3514          ::*****
3515          ;*INCREMENT THE PASS NUMBER ($PASS)
3516          ;*INDICATE END-OF-PROGRAM AFTER 8. PASSES THRU THE PROGRAM
3517          ;*IF THERES A MONITOR GO TO IT
3518          ;*IF THERE ISN'T JUMP TO RESTART
3519
3520          SEOP:
3521          TYPE     655 ;TYPE ASCIZ STRING
3522          BR      645 ;GET OVER THE ASCIZ
3523          ::655:  .ASCIZ <CR><LF><LF>/END OF PASS/
3524          645:
3525          TST     2#DRVSEL ;ANY DRIVES SELECTED?
3526          BEQ     15 ;NO--BRANCH
3527          TYPE     675 ;TYPE ASCIZ STRING
3528          BR      665 ;GET OVER THE ASCIZ
3529          ::675:  .ASCIZ / ON DRIVE/
3530          665:
3531          MOV     2#CHKDRV, -(SP) ;SAVE 2#CHKDRV FOR TYPEOUT
3532          TYPOS   ;GO TYPE--OCTAL ASCII
3533          .BYTE  2 ;TYPE 2 DIGIT(S)
3534          .BYTE  0 ;SUPPRESS LEADING ZEROS
3535          TYPE     ,695 ;TYPE ASCIZ STRING
3536
3537          017206 000240
3538          017210 033737 001430 001236
3539          017216 001002
3540          017220 000137 017356
3541          017224 012737 000022 001102 645:
3542          017232 004737 024560 JSR PC.LODPRM
3543          017236 012737 017274 001110 MOV #TEST22, 2#SLPERR
3544          017244 013777 001102 161670 MOV STSTNM, 2#DISPLAY
3545          017252 013737 001506 001204 MOV 2#RPT, $TIMES
3546          017260 112737 000144 001115 MOV #100, 2#SERMAX
3547          017266 012737 017274 001106 MOV #TEST22, 2#SLPADR
3548          017274 012706 001100 TEST22: MOV #STACK, SP
3549          017300 013737 001512 004116 MOV LC, DPB.A+12
3550          017306 112737 000105 004106 MOV #SEEK, 2#DPB.A+2
3551          017314 004037 025002 JSR RD, 2#CALL.A
3552          017320 004037 026334 JSR RD, STALL
3553          017324 001360 .WORD STALL3
3554          017326 013737 001510 004116 MOV FC, DPB.A+12
3555          017334 112737 000105 004106 MOV #SEEK, 2#DPB.A+2
3556          017342 004037 025002 JSR RD, 2#CALL.A
3557          017346 004037 026334 JSR RD, STALL
3558          017352 001360 .WORD STALL3
3559          017354 000004 EXIT22: SCOPE
    
```


3538	017446	000412			BR	685	::GET OVER THE ASCIZ
3539					::695: .ASCIZ	/	ERRORS DETECTED=
3540	017474				MOV	2#SERTTL,-(SP)	::SAVE 2#SERTTL FOR TYPEOUT
3541	017474	013746	001112		TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
3542	017500	104402			15: CLR	2#SERTTL	::ZERO ERROR TOTAL
3543	017502	005037	001112		CLR	STSTNM	::ZERO THE TEST NUMBER
3544	017506	005037	001102		CLR	STIMES	::ZERO THE NUMBER OF ITERATIONS
3545	017512	005037	001204		INC	SPASS	::INCREMENT THE PASS NUMBER
3546	017516	005237	001100		BIC	#10000,SPASS	::DON'T ALLOW A NEG. NUMBER
3547	017522	042737	100000	001100	DEC	(PC)+	::LOOP?
3548	017530	005327			SEOPCT: .WORD	B.	
3549	017532	000010			BGT	SDOAGN	::YES
3550	017534	003027			MOV	(PC)+,2(PC)+	::RESTORE COUNTER
3551	017536	012737			SENOCT: .WORD	B.	
3552	017540	000010			SEOPCT		
3553	017542	017532			TYPE	655	::TYPE ASCIZ STRING
3554	017544	104401	017552		BR	645	::GET OVER THE ASCIZ
3555	017550	000407			::655: .ASCIZ	<CR><LF>/END OF TEST/	
3556	017570				645: TYPE	SENULL	::TYPE NULL CHARACTER
3557	017570	104401	017620		SGET42: MOV	2#42,R0	::GET MONITOR ADDRESS
3558	017570	013700	000042		BEQ	SDOAGN	::BRANCH IF NO MONITOR
3559	017600	001405			RESET		::CLEAR THE WORLD
3560	017602	000005			SENDAD: JSR	PC,(R0)	::GO TO MONITOR
3561	017604	004710			NOP		::SAVE ROOM
3562	017606	000240			NOP		::FOR
3563	017610	000240			NOP		::ACT11
3564	017612	000240			SDOAGN:		
3565	017614				JMP	2(PC)+	::RETURN
3566	017614	000137			SRTNAD: .WORD	RESTART	
3567	017616	006072			SENULL: .BYTE	-1,-1,0	::NULL CHARACTER STRING
3568	017620	377	377	000	.EVEN		
3570	017624						

3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626

017624
017624 104407
017626 032777 000400 161304
017634 001411
017636 005737 001230
017642 001406
017644 013737 001420 001150
017652 013737 001422 001152
017660 105237 001103
017664 001775
017666 013777 001102 161246
017674 032777 002000 161236
017702 001402
017704 104401 001210
017710 005237 001112
017714 011637 001116
017720 162737 000002 001116
017726 117737 161164 001114
017734 032777 020000 161176
017742 001004
017744 004737 020032
017750 104401 001215
017754
017754 005777 161160
017760 100002
017762 000000
017764 104407
017766 032777 001000 161144
017774 001402
017776 013716 001110
020002 005737 001206
020006 001402
020010 013716 001206
020014
020014 013737 001414 001150
020022 013737 001416 001152
020030 000002

.SBTTL *** SYSMAC SUBROUTINES ***

.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 TYPEERR 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
*SW09=1 LOOP ON ERROR
*CALL
* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

SERROR:

CKSWR
BIT #SW08, @SWR
BEQ 7S
TST @LPTAVL
BEQ 7S
MOV @LPS, @STPS
MOV @LPB, @STPB
7S: INCB SERFLG
BEQ 7S
MOV \$STNM, @DISPLAY
BIT @BIT10, @SWR
BEQ 1S
TYPE \$BELL
1S: INC \$ERTTL
MOV (SP), \$ERRPC
SUB #2, \$ERRPC
MOVB @SERAPC, \$ITEMB
BIT @BIT13, @SWR
BNE 20S
JSR PC, TYPEERR
TYPE , \$CALF
20S:
2S: TST @SWR
BPL 3S
HALT
CKSWR
3S: BIT @BIT09, @SWR
BEQ 4S
MOV \$LPERA, (SP)
4S: TST \$ESCAPE
BEQ 5S
MOV \$ESCAPE, (SP)
5S: MOV @TPS, @STPS
MOV @TPB, @STPB
RTI

;; TEST FOR CHANGE IN SOFT-SWR
;; SEND ERROR MESSAGE TO TTY?
;; YES--BRANCH
;; IS THERE A LINE PRINTER AVAILABLE?
;; NO--BRANCH
;; YES--SETUP STATUS
;; AND BUFFER REG.'S FOR LINE PRINTER
;; SET THE ERROR FLAG
;; DON'T LET THE FLAG GO TO ZERO
;; DISPLAY TEST NUMBER AND ERROR FLAG
;; BELL ON ERROR?
;; NO - SKIP
;; RING BELL
;; COUNT THE NUMBER OF ERRORS
;; GET ADDRESS OF ERROR INSTRUCTION
;; STRIP AND SAVE THE ERROR ITEM CODE
;; SKIP TYPEOUT IF SET
;; SKIP TYPEOUTS
;; GO TO USER ERROR ROUTINE
;; HALT ON ERROR
;; SKIP IF CONTINUE
;; HALT ON ERROR!
;; TEST FOR CHANGE IN SOFT-SWR
;; LOOP ON ERROR SWITCH SET?
;; BR IF NO
;; FUDGE RETURN FOR LOOPING
;; CHECK FOR AN ESCAPE ADDRESS
;; BR IF NONE
;; FUDGE RETURN ADDRESS FOR ESCAPE
;; SET STATUS AND BUFFER REG.'S
;; FOR TTY
;; RETURN FROM ERROR CALL

3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682

020032 113737 001102 001176
020040 104412
020042 162700 000004
020046 010037 001162
020052 010137 001164
020056 010237 001166
020062 010337 001170
020066 010437 001172
020072 010537 001174
020076 113700 001114
020102 010001
020104 005300
020106 106300
020110 106300
020112 106300
020114 103002
020116 062700 000240
020122 062700 004306
020126 012037 020142
020132 001447
020134 104401 001215
020140 104401
020142 000000
020144 162701 000041
020150 100440
020152 013701 001260
020156 106301
020160 006301
020162 012702 004254
020166 005003
020170 104401 020176
020174 000402
020202
020202 012237 020224
020206 006301
020210 103013
020212 005103
020214 001002
020216 104401 001215
020222 104401
020224 000000
020226 005701
020230 001403
020232 104401 044010
020236 000761
020240 001360

```
.SBTTL TYPERR - TYPE ERROR ROUTINE
;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
;WHICH ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR
;TABLE" ($ERRTB), AND REPORTS THE APPROPRIATE INFORMATION
;CONCERNING THE ERROR.
CALL
;JSR PC,@#TYPERR
;RETURN
TYPERR: MOVB @#STSTNM,@#STMPO ;SAVE THE TEST NUMBER
;SAVREG ;SAVE R0 - R5
;SUB @4,R0 ;FORM TEST PC
;MOV R0,@#SREG0 ;COPY R0-R5 IN SREG0-SREG5
;MOV R1,@#SREG1
;MOV R2,@#SREG2
;MOV R3,@#SREG3
;MOV R4,@#SREG4
;MOV R5,@#SREG5
;MOVB @#ITEMB,R0 ;PICKUP ERROR ITEM NUMBER
;MOV R0,R1 ;AND COPY IT INTO R1
;DEC R0 ;FORM INDEX FOR ERROR TABLE
;ASLB R0
;ASLB R0
;ASLB R0
;BCC 1$ ;IS ERROR > 37?
;ADD @ITEM#1-$ERRTB,R0 ;YES--FORM OFFSET
;ADD @ERRTB,R0 ;FORM ADDRESS
;MOV (R0)+,@# ;GET ERROR MESSAGE (EM) POINTER
;BEQ 7$ ;BRANCH IF THERE ISN'T ONE
;TYPE ,$CRLF ;"CARRIAGE RETURN - LINE FEED"
;TYPE
;WORD 0 ;"EM" POINTER GOES HERE
;SUB @41,R1 ;SPECIAL ERROR ITEM NUMBER?
;BHI 7$ ;NO--BRANCH
;MOV @#SVSTAT,R1 ;GET STATUS/ERROR INDICATOR
;ASLB R1 ;STRIP "DONE" BIT (BIT07)
;ASL R1 ;STRIP "ERROR" BIT (BIT15)
;MOV @STATBL,R2 ;1ST ADDRESS ON STATUS MESSAGE POINTERS
;CLR R3 ;CARRIAGE RETURN-LINE FEED SWITCH
;TYPE ,65$ ;TYPE ASCIZ STRING
;BR 64$ ;GET OVER THE ASCIZ
;65$: .ASCIZ / (/
;64$:
;3$: MOV (R2)+,@# ;MESSAGE POINTER
;ASL R1 ;TYPE THIS MESSAGE?
;BCC 6$ ;NO--BRANCH
;COM R3 ;YES--TYPE A "CR" & "LF"?
;BNE 4$ ;NO--BRANCH
;TYPE ,$CRLF ;YES
;4$:
;5$: .WORD 0 ;MESSAGE POINTER GOES HERE
;TST R1 ;MORE TO TYPE?
;BEQ 6$ ;NO--BRANCH
;TYPE ,MSG.SP ;YES--SPACES
;BR 3$ ;LOOP
;6$: BNE 3$ ;BRANCH IF NOT FINISHED
```


3683	020242	104401	020250		TYPE	67S	::TYPE ASCIZ STRING
3684	020246	000401			BR	66S	::GET OVER THE ASCIZ
3685				::67S:	.ASCIZ	/)/	
3686	020252			66S:			
3687	020252	012037	020266	7S:	MOV	(R0)+,8S	:PICK UP DATA HEADER (DH) POINTER
3688	020256	001404			BEQ	9S	:BRANCH IF NONE
3689	020260	104401	001215		TYPE	,SCRLF	:CARRIAGE RETURN-LINE FEED
3690	020264	104401			TYPE		
3691	020266	000000		8S:	.WORD	0	:"DH" POINTER GOES HERE
3692	020270	012001		9S:	MOV	(R0)+,R1	:PICKUP DATA TABLE (DT) POINTER
3693	020272	001450			BEQ	20S	:BRANCH IF NONE
3694	020274	005005			CLR	R5	:SET INDENT SWITCH
3695	020276	012000			MOV	(R0)+,R0	:DATA FORMAT (DF) POINTER
3696	020300	012002			MOV	(R0)+,R2	:NUMBER OF DH'S TO TYPE
3697	020302	001441			BEQ	17S	:BRANCH IF DH NUMBER IS 0
3698	020304	005105			COM	R5	:NO INDENT
3699	020306	104401	001215		TYPE	,SCRLF	:CARRIAGE RETURN-LINE FEED
3700	020312	112003		10S:	MOVB	(R0)+,R3	:NUMBER OF DATA WORDS TO TYPE
3701	020314	112004			MOVB	(R0)+,R4	:AND HOW TO TYPE THEM
3702	020316	006004		11S:	ROR	R4	:OCTAL OR DECIMAL?
3703	020320	103403			BCS	12S	:DECIMAL--BRANCH
3704	020322	013146			MOV	2(R1)+,-(SP)	:SAVE 2(R1)+ FOR TYPEOUT
3705	020324	104402			TYPDC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
3706	020326	000402			BR	13S	
3707	020330			12S:			
3708	020330	013146			MOV	2(R1)+,-(SP)	:SAVE 2(R1)+ FOR TYPEOUT
3709	020332	104405			TYPDS		:GO TYPE--DECIMAL ASCII WITH SIGN
3710	020334	005303		13S:	DEC	R3	:MORE NUMBERS TO TYPE?
3711	020336	001403			BEQ	14S	:NO--BRANCH
3712	020340	104401	044010		TYPE	,MSG.SP	:YES--TYPE SEPERATORS
3713	020344	000764			BR	11S	:LOOP
3714	020346	005302		14S:	DEC	R2	:MORE DH'S?
3715	020350	003421			BLE	20S	:NO--BRANCH
3716	020352	104401	001215		TYPE	,SCRLF	:YES--START A NEW LINE
3717	020356	005105			COM	R5	:INDENT?
3718	020360	001002			BNE	15S	:NO--BRANCH
3719	020362	104401	044010		TYPE	,MSG.SP	:YES--TYPE SPACES
3720	020366	012037	020374	15S:	MOV	(R0)+,16S	:GET NEXT DH
3721	020372	104401			TYPE		:AND TYPE IT
3722	020374	000000		16S:	.WORD	0	:DH POINTER GOES HERE
3723	020376	104401	001215		TYPE	,SCRLF	:CARRIAGE RETURN-LINE FEED
3724	020402	005705			TST	R5	:INDENT?
3725	020404	001342			BNE	10S	:NO--BRANCH
3726	020406	104401	044010	17S:	TYPE	,MSG.SP	:YES--TYPE SPACES
3727	020412	000737			BR	10S	:LOOP
3728	020414	104413		20S:	RESREG		:RESTORE R0 - R5
3729	020416	000207			RTS	PC	:RETURN

.SBTTL TYPE ROUTINE

```

:*****
:ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
:THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
:NOTE1: SNULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
:NOTE2: SFILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
:NOTE3: SFILLC CONTAINS THE CHARACTER TO FILL AFTER.

```

3730
3731
3732
3733
3734
3735
3736
3737
3738


```

3739
3740
3741
3742
3743
3744
3745
3746
3747
3748 020420 105737 001157
3749 020424 100002
3750 020426 000000
3751 020430 000407
3752 020432 010046
3753 020434 017600 000002
3754 020440 112046
3755 020442 001005
3756 020444 005726
3757 020446 012600
3758 020450 062716 000002
3759 020454 000002
3760 020456 122716 000011
3761 020462 001430
3762 020464 122716 000200
3763 020470 001006
3764 020472 005726
3765 020474 104401
3766 020476 001215
3767 020500 105037 020634
3768 020504 000755
3769 020506 004737 020570
3770 020512 123726 001156
3771 020516 001350
3772 020520 013746 001154
3773
3774 020524 105366 000001
3775 020530 002770
3776 020532 004737 020570
3777 020536 105337 020634
3778 020542 000770
3779
3780
3781
3782 020544 112716 000040
3783 020550 004737 020570
3784 020554 132737 000007 020634
3785 020562 001372
3786 020564 005726
3787 020566 000724
3788 020570 105777 160354
3789 020574 100375
3790 020576 116677 000002 160346
3791 020604 122766 000015 000002
3792 020612 001003
3793 020614 105037 020634
3794 020620 000406

;HORIZONTAL TAB PROCESSOR
8S:  MOV  #'(SP)
9S:  JSR  PC,STYPC
      BITB #7,SCHARCNT
      BNE 9S
      TST (SP)+
      BR 2S
STYPC: TST 2STPS
        BPL STYPC
        MOVB 2(SP),2STPB
        CMPB #CR,2(SP)
        BNE 1S
        CLRB SCHARCNT
        BR STYPC

;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;; IS THERE A TERMINAL?
;; BR IF YES
;; HALT HERE IF NO TERMINAL
;; LEAVE
;; SAVE R0
;; GET ADDRESS OF ASCIZ STRING
;; PUSH CHARACTER TO BE TYPED ONTO STACK
;; BR IF IT ISN'T THE TERMINATOR
;; IF TERMINATOR POP IT OFF THE STACK
;; RESTORE R0
;; ADJUST RETURN PC
;; RETURN
;; BRANCH IF <HT>
;; BRANCH IF NOT <CR LF>
;; POP <CR><LF> EQUIV
;; TYPE A CR AND LF
;; CLEAR CHARACTER COUNT
;; GET NEXT CHARACTER
;; GO TYPE THIS CHARACTER
;; IS IT TIME FOR FILLER CHARS.?
;; IF NO GO GET NEXT CHAR.
;; GET # OF FILLER CHARS. NEEDED
;; AND THE NULL CHAR.
;; DOES A NULL NEED TO BE TYPED?
;; BR IF NO--GO POP THE NULL OFF OF STACK
;; GO TYPE A NULL
;; DO NOT COUNT AS A COUNT
;; LOOP
    
```


3795 020622 122766 000012 000002
3796 020630 001402
3797 020632 105227
3798 020634 000000
3799 020636 000207
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827 020640 017646 000000
3828 020644 116637 000001 021063
3829 020652 112637 021065
3830 020656 062716 000002
3831 020662 000406
3832 020664 112737 000001 021063
3833 020672 112737 000006 021065
3834 020700 112737 000005 021062
3835 020706 010346
3836 020710 010446
3837 020712 010546
3838 020714 113704 021065
3839 020720 005404
3840 020722 062704 000006
3841 020726 110437 021064
3842 020732 113704 021063
3843 020736 016605 000012
3844 020742 005003
3845 020744 006105
3846 020746 000404
3847 020750 006105
3848 020752 006105
3849 020754 006105
3850 020756 010503

15: CMPB #LF,2(SP) ;: IS CHARACTER A LINE FEED?
BEQ STYPEX ;: BRANCH IF YES
INCB (PC)+ ;: COUNT THE CHARACTER
SCHARCNT: .WORD 0 ;: CHARACTER COUNT STORAGE
STYPEX: 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.

: *STYPOS---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

: *STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST

: *STYPOS OR STYPOC

: *CALL:

: * MOV NUM,-(SP) ;: NUMBER TO BE TYPED
: * TYPON ;: CALL FOR TYPEOUT

: *STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

: *CALL:

: * MOV NUM,-(SP) ;: NUMBER TO BE TYPED
: * TYPOC ;: CALL FOR TYPEOUT

STYPOS: MOV 2(SP),-(SP) ;: PICKUP THE MODE
MOV 1(SP),SOFILL ;: LOAD ZERO FILL SWITCH
MOV (SP)+,SOMODE+1 ;: NUMBER OF DIGITS TO TYPE
ADD #2,(SP) ;: ADJUST RETURN ADDRESS
BR STYPON

STYPOC: MOV #1,SOFILL ;: SET THE ZERO FILL SWITCH
MOV #6,SOMODE+1 ;: SET FOR SIX(6) DIGITS

STYPON: MOV #5,SOCNT ;: SET THE ITERATION COUNT
MOV R3,-(SP) ;: SAVE R3
MOV R4,-(SP) ;: SAVE R4
MOV R5,-(SP) ;: SAVE R5
MOV #SOMODE+1,R4 ;: GET THE NUMBER OF DIGITS TO TYPE
NEG R4

ADD #6,R4 ;: SUBTRACT IT FOR MAX. ALLOWED
MOV R4,SOMODE ;: SAVE IT FOR USE

MOV SOFILL,R4 ;: GET THE ZERO FILL SWITCH
MOV 12(SP),R5 ;: PICKUP THE INPUT NUMBER
CLR R3 ;: CLEAR THE OUTPUT WORD

15: ROL R5 ;: ROTATE MSB INTO "C"

BR 38 ;: GO DO MSB

25: ROL R5 ;: FORM THIS DIGIT

ROL R5
ROL R5
MOV R5,R3


```

3851 020760 006103          3S:  ROL  R3          :: GET LSB OF THIS DIGIT
3852 020762 105337 021064    DECB $OMODE      :: TYPE THIS DIGIT?
3853 020766 100016          BPL  7$          :: BR IF NO
3854 020770 042703 177770    BIC  #177770,R3  :: GET RID OF JUNK
3855 020774 001002          BNE  4$          :: TEST FOR 0
3856 020776 005704          TST  R4          :: SUPPRESS THIS 0?
3857 021000 001403          BEQ  5$          :: BR IF YES
3858 021002 005204          4S:  INC  R4          :: DON'T SUPPRESS ANYMORE 0'S
3859 021004 052703 000060    BIS  #'0,R3      :: MAKE THIS DIGIT ASCII
3860 021010 052703 000040    5S:  BIS  #' ,R3   :: MAKE ASCII IF NOT ALREADY
3861 021014 110337 021060    MOVB R3,B$       :: SAVE FOR TYPING
3862 021020 104401 021060    TYPE B$          :: GO TYPE THIS DIGIT
3863 021024 105337 021062    7S:  DECB $OCNT   :: COUNT BY 1
3864 021030 003347          BGT  2$          :: BR IF MORE TO DO
3865 021032 002402          BLT  6$          :: BR IF DONE
3866 021034 075204          INC  R4          :: INSURE LAST DIGIT ISN'T A BLANK
3867 021036 000744          BR   2$          :: GO DO THE LAST DIGIT
3868 021040 012605          6S:  MOV  (SP)+,R5  :: RESTORE R5
3869 021042 012604          MOV  (SP)+,R4     :: RESTORE R4
3870 021044 012603          MOV  (SP)+,R3     :: RESTORE R3
3871 021046 016666 000002 000004  MOV  2(SP),4(SP)  :: SET THE STACK FOR RETURNING
3872 021054 012616          MOV  (SP)+,(SP)   ::
3873 021056 000002          RTI              :: RETURN
3874 021060 000          8S:  .BYTE 0      :: STORAGE FOR ASCII DIGIT
3875 021061 000          .BYTE 0          :: TERMINATOR FOR TYPE ROUTINE
3876 021062 000          $OCNT: .BYTE 0   :: OCTAL DIGIT COUNTER
3877 021063 000          $OFILL: .BYTE 0  :: ZERO FILL SWITCH
3878 021064 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

STYPDS:
MOV R0,-(SP)          :: PUSH R0 ON STACK
MOV R1,-(SP)          :: PUSH R1 ON STACK
MOV R2,-(SP)          :: PUSH R2 ON STACK
MOV R3,-(SP)          :: PUSH R3 ON STACK
MOV R5,-(SP)          :: PUSH R5 ON STACK
MOV #20200,-(SP)      :: SET BLANK SWITCH AND SIGN
MOV 20(SP),R5         :: GET THE INPUT NUMBER
BPL 1$               :: BR IF INPUT IS POS.
NEG R5                :: MAKE THE BINARY NUMBER POS.
MOVB #'-,1(SP)       :: MAKE THE ASCII NUMBER NEG.
1S:  CLR R0           :: ZERO THE CONSTANTS INDEX
MOV #DBLK,R3         :: SETUP THE OUTPUT POINTER
MOVB #' ,(R3)+       :: SET THE FIRST CHARACTER TO A BLANK
2S:  CLR R2           :: CLEAR THE BCD NUMBER
    
```


MO6

```

3907 021136 016001 021272          MOV      SDBL(R0),R1      ;; GET THE CONSTANT
3908 021142 160105          3$: SUB      R1,R5        ;; FORM THIS BCD DIGIT
3909 021144 002402          BLT      4$,R5          ;; BR IF DONE
3910 021146 005202          INC      R2            ;; INCREASE THE BCD DIGIT BY 1
3911 021150 000774          BR       3$
3912 021152 060105          4$: ADD      R1,R5        ;; ADD BACK THE CONSTANT
3913 021154 005702          TST      R2            ;; CHECK IF BCD DIGIT=0
3914 021156 001002          BNE      5$            ;; FALL THROUGH IF 0
3915 021160 105716          TSTB     (SP)          ;; STILL DOING LEADING 0'S?
3916 021162 100407          BMI      7$            ;; BR IF YES
3917 021164 106316          5$: ASLB     (SP)          ;; MSD?
3918 021166 103003          BCC      6$            ;; BR IF NO
3919 021170 116663 000001 177777  MOVB     1(SP),-1(R3)    ;; YES--SET THE SIGN
3920 021176 052702 000060 6$: BIS      #'0,R2      ;; MAKE THE BCD DIGIT ASCII
3921 021202 052702 000040 7$: BIS      #' ,R2      ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
3922 021206 110223          MOVB     R2,(R3)+      ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
3923 021210 005720          TST      (R0)+        ;; JUST INCREMENTING
3924 021212 020027 000010  CMP      R0,#10        ;; CHECK THE TABLE INDEX
3925 021216 002746          BLT      2$            ;; GO DO THE NEXT DIGIT
3926 021220 003002          BGT      8$            ;; GO TO EXIT
3927 021222 010502          MOV      R5,R2        ;; GET THE LSD
3928 021224 000764          BR       6$            ;; GO CHANGE TO ASCII
3929 021226 105726          8$: TSTB     (SP)+      ;; WAS THE LSD THE FIRST NON-ZERO?
3930 021230 100003          BPL      9$            ;; BR IF NO
3931 021232 116663 177777 177776 9$: MOVB     -1(SP),-2(R3) ;; YES--SET THE SIGN FOR TYPING
3932 021240 105013          CLRB     (R3)          ;; SET THE TERMINATOR
3933 021242 012605          MOV      (SP)+,R5      ;; POP STACK INTO R5
3934 021244 012603          MOV      (SP)+,R3      ;; POP STACK INTO R3
3935 021246 012602          MOV      (SP)+,R2      ;; POP STACK INTO R2
3936 021250 012601          MOV      (SP)+,R1      ;; POP STACK INTO R1
3937 021252 012600          MOV      (SP)+,R0      ;; POP STACK INTO R0
3938 021254 104401 021302          TYPE     SDBLK         ;; NOW TYPE THE NUMBER
3939 021260 016666 000002 000004  MOV      2(SP),4(SP)    ;; ADJUST THE STACK
3940 021266 012616          MOV      (SP)+,(SP)
3941 021270 000002          RTI
3942 021272 023420          SDBL: 10000.
3943 021274 001750          1000.
3944 021276 000144          100.
3945 021300 000012          10.
3946 021302 000004          SDBLK: .BLKW 4
3947
3948          .SBTTL TTY INPUT ROUTINE
3949
3950          ;; *****
3951          .ENABL LSB
3952 021312 000000  STKCNT: .WORD 0        ;; NUMBER OF ITEMS IN QUEUE
3953 021314 000000  STKQIN: .WORD 0        ;; INPUT POINTER
3954 021316 000000  STKQOUT: .WORD 0       ;; OUTPUT POINTER
3955 021320 000002  STKQSRT: .BLKB 2       ;; TTY KEYBOARD QUEUE
3956 021322
3957
3958          ;*TK INITIALIZE ROUTINE
3959          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
3960          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
3961
3962          ;*CALL:
  
```



```

3963          : *      JSR      PC,STKINT
3964          : *      RETURN
3965
3966 021322 005037 021312          :STKINT: CLR      STKCNT          : CLEAR COUNT OF ITEMS IN QUEUE
3967 021326 012737 021320 021314 :      MOV      #STKQSR,STKQIN : MOVE THE STARTING ADDRESS OF THE
3968 021334 013737 021314 021316 :      MOV      STKQIN,STKQOUT : QUEUE INTO THE INPUT & OUTPUT POINTERS.
3969 021342 012737 021372 000060 :      MOV      #STKSRV,#STKVEC : INITIALIZE THE KEYBOARD VECTOR
3970 021350 012737 000200 000062 :      MOV      #200,#STKVEC+2 : "BR" LEVEL 4
3971 021356 005777 157564          :      TST      #STKB          : CLEAR DONE FLAG
3972 021362 012777 000100 157554 :      MOV      #100,#STKS      : ENABLE TTY KEYBOARD INTERRUPT
3973 021370 000207          :      RTS      PC          : RETURN TO CALLER
3974
3975          : *TK SERVICE ROUTINE
3976          : *THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
3977          : *BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
3978          : *IT IN THE QUEUE.
3979          : *IF THE CHARACTER IS A "CONTROL-C" (↑C) STKINT IS CALLED AND
3980          : *UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START2)
3981
3982 021372 117746 157550          :STKSRV: MOVVB   #STKB, -(SP)      : PICKUP THE CHARACTER
3983 021376 042716 177600          :      BIC      #↑C177, (SP)      : STRIP THE JUNK
3984 021402 021627 000003          :      CMP      (SP), #3          : IS IT A CONTROL C?
3985 021406 001007          :      BNE      1$              : BRANCH IF NO
3986 021410 104401 022522          :      TYPE    ,SCNTLC          : TYPE A CONTROL-C (↑C)
3987 021414 004737 021322          :      JSR      PC,STKINT        : INIT THE KEYBOARD
3988 021420 005726          :      TST      (SP)+          : CLEAN UP STACK
3989 021422 000137 004660          :      JMP      START2          : CONTROL C RESTART
3990 021426 021627 000007          :      CMP      (SP), #7          : IS IT A CONTROL G?
3991 021432 001004          :      BNE      2$              : BRANCH IF NO
3992 021434 022737 000176 001140 :      CMP      #SWREG,SWR        : IS SOFT-SWR SELECTED?
3993 021442 001500          :      BEQ     6$              : GO TO SWR CHANGE
3994
3995          :      2$:
3996 021444 022737 000002 021312 :      CMP      #2,STKCNT        : IS THE QUEUE FULL?
3997 021452 001004          :      BNE      3$              : BRANCH IF NO
3998 021454 104401 001210          :      TYPE    ,SBELL          : RING THE TTY BELL
3999 021460 005726          :      TST      (SP)+          : CLEAN CHARACTER OFF OF STACK
4000 021462 000451          :      BR      5$              : EXIT
4001 021464 021627 000023          :      3$: CMP      (SP), #23      : IS IT A CONTROL-S?
4002 021470 001021          :      BNE      32$           : BRANCH IF NO
4003 021472 005077 157446          :      CLR      #STKS          : DISABLE TTY KEYBOARD INTERRUPTS
4004 021476 005726          :      TST      (SP)+          : CLEAN CHAR OFF STACK
4005 021500 105777 157440          :      31$: TSTB    #STKS          : WAIT FOR A CHAR
4006 021504 100375          :      BPL     31$           : LOOP UNTIL ITS THERE
4007 021506 117746 157434          :      MOVVB   #STKB, -(SP)      : GET THE CHARACTER
4008 021512 042716 177600          :      BIC      #↑C177, (SP)      : MAKE IT 7-BIT ASCII
4009 021516 022627 000021          :      CMP      (SP)+, #21      : IS IT A CONTROL-Q?
4010 021522 001366          :      BNE      31$           : BRANCH IF NO
4011 021524 012777 000100 157412 :      MOV      #100,#STKS      : REENABLE TTY KEYBOARD INTERRUPTS
4012 021532 000002          :      RTI     #STKS          : RETURN
4013 021534 005237 021312          :      32$: INC      STKCNT        : COUNT THIS CHARACTER
4014 021540 021627 000140          :      CMP      (SP), #140      : IS IT UPPER CASE?
4015 021544 002405          :      BLT     4$              : BRANCH IF YES
4016 021546 021627 000175          :      CMP      (SP), #175      : IS IT A SPECIAL CHAR?
4017 021552 003002          :      BGT     4$              : BRANCH IF YES
4018 021554 042716 000040          :      BIC      #40, (SP)        : MAKE IT UPPER CASE
    
```


DU

```

4019 021560 112677 177530      4S:  MOVB  (SP)+,STKQIN  ;;AND PUT IT IN QUEUE
4020 021564 005237 021314      INC    STKQIN        ;;UPDATE THE POINTER
4021 021570 023727 021314 021322  CMP    STKQIN,#STKGEND ;;GO OFF THE END?
4022 021576 001003                BNE    5S           ;;BRANCH IF NO
4023 021600 012737 021320 021314  MOV    #STKQSR,STKQIN ;;RESET THE POINTER
4024 021606 000002                SS:  RTI             ;;RETURN
4025
4026                ;;*****
4027                ;;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
4028                ;;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4029                ;;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
4030                ;;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
4031 021610 022737 000176 001140  $CKSMR: CMP    #SMREG,SMR  ;;IS THE SOFT-SMR SELECTED
4032 021616 001124                BNE    15S         ;;EXIT IF NOT
4033 021620 105777 157320                TSTB  #STKS        ;;IS A CHAR WAITING?
4034 021624 100121                BPL    15S         ;;IF NOT, EXIT
4035 021626 117746 157314                MOVB  #STKB,-(SP)   ;;YES
4036 021632 042716 177600                BIC  #1C177,(SP)  ;;MAKE IT 7-BIT ASCII
4037 021636 021627 000007                CMP    (SP),#7     ;;IS IT A CONTROL-G?
4038 021642 001300                BNE    2S         ;;IF NOT, PUT IT IN THE TTY QUEUE
4039
4040                ;;AND EXIT
4041
4042                ;;*****
4043                ;;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
4044                ;;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
4045                ;;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
4046 021644 123727 001134 000001  6S:  CMPS  $AUTOB,#1  ;;ARE WE RUNNING IN AUTO-MODE?
4047 021652 001674                BEQ    2S         ;;BRANCH IF YES
4048 021654 005726                TST  (SP)+        ;;CLEAR CONTROL-G OFF STACK
4049 021656 004737 021322                JSR  PC,STKINT    ;;FLUSH THE TTY INPUT QUEUE
4050 021662 005077 157256                CLR  #STKS        ;;DISABLE TTY KEYBOARD INTERRUPTS
4051 021666 112737 000001 001135  MOVB  #1,SINTAG   ;;SET INTERRUPT MODE INDICATOR
4052
4053                ;;ECHO THE CONTROL-G (#G)
4054 021674 104401 022534                TYPE  ,SCNTLG     ;;
4055 021700 104401 022541                TYPE  ,SMSMR      ;;TYPE CURRENT CONTENTS
4056 021704 013746 000176                MOV  $SMREG,-(SP) ;;SAVE SMREG FOR TYPEOUT
4057 021710 104402                TYPEOC           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4058 021712 104401 022552                TYPE  ,SMNEW      ;;PROMPT FOR NEW SMR
4059 021716 005046                19S: CLR  -(SP)    ;;CLEAR COUNTER
4060 021720 005046                CLR  -(SP)        ;;THE NEW SMR
4061 021722 105777 157216                7S:  TSTB  #STKS    ;;CHAR THERE?
4062 021726 100375                BPL  7S           ;;IF NOT TRY AGAIN
4063
4064                ;;PICK UP CHAR
4065 021730 117746 157212                MOVB  #STKB,-(SP) ;;
4066 021734 042716 177600                BIC  #1C177,(SP) ;;MAKE IT 7-BIT ASCII
4067
4068                ;;IS IT A CONTROL-C?
4069 021740 021627 000003                CMP  (SP),#3     ;;
4070 021744 001015                BNE  9S          ;;BRANCH IF NOT
4071 021746 104401 022522                TYPE  ,SCNTLC     ;;YES, ECHO CONTROL-C (#C)
4072 021752 062706 000006                ADD  #6,SP        ;;CLEAN UP STACK
4073 021756 123727 001135 000001  CMPS  $SINTAG,#1  ;;REENABLE TTY KEYBOARD INTERRUPTS?
4074 021764 001003                BNE  8S          ;;BRANCH IF NO
4075 021766 012777 000100 157150  MOV  #100,#STKS  ;;ALLOW TTY KEYBOARD INTERRUPTS
4076 021774 000137 004660                8S:  JMP  START2   ;;CONTROL-C RESTART
    
```



```

4075 022000 021627 000025 95:  CMP (SP),#25      ;; IS IT A CONTROL-U?
4076 022004 001005          BNE 10$          ;; BRANCH IF NOT
4077 022006 104401 022527      TYPE ,SCNTLU     ;; YES, ECHO CONTROL-U (↑U)
4078 022012 062706 000006 20$:  ADD #6,SP       ;; IGNORE PREVIOUS INPUT
4079 022016 000737          BR 19$          ;; LET'S TRY IT AGAIN
4080
4081
4082 022020 021627 000015 10$:  CMP (SP),#15     ;; IS IT A <CR>?
4083 022024 001022          BNE 16$          ;; BRANCH IF NO
4084 022026 005766 000004      TST 4(SP)        ;; YES, IS IT THE FIRST CHAR?
4085 022032 001403          BEQ 11$          ;; BRANCH IF YES
4086 022034 016677 000002 157076  MOV 2(SP),@SWR   ;; SAVE NEW SWR
4087 022042 062706 000006 11$:  ADD #6,SP       ;; CLEAR UP STACK
4088 022046 104401 001215 14$:  TYPE ,SCRLF     ;; ECHO <CR> AND <LF>
4089 022052 123727 001135 000001  CMPB $INTAG,#1  ;; RE-ENABLE TTY KBD INTERRUPTS?
4090 022060 001003          BNE 15$          ;; BRANCH IF NOT
4091 022062 012777 000100 157054  MOV #100,@STKS  ;; RE-ENABLE TTY KBD INTERRUPTS
4092 022070 000002          RTI                    ;; RETURN
4093 022072 004737 020570 16$:  JSR PC,$TYPEC   ;; ECHO CHAR
4094 022076 021627 000060      CMP (SP),#60    ;; CHAR < 0?
4095 022102 002420          BLT 18$          ;; BRANCH IF YES
4096 022104 021627 000067      CMP (SP),#67    ;; CHAR > 7?
4097 022110 003015          BGT 18$          ;; BRANCH IF YES
4098 022112 042726 000060      BIC #60,(SP)+   ;; STRIP-OFF ASCII
4099 022116 005766 000002      TST 2(SP)       ;; IS THIS THE FIRST CHAR
4100 022122 001403          BEQ 17$          ;; BRANCH IF YES
4101 022124 006316          ASL (SP)        ;; NO, SHIFT PRESENT
4102 022126 006316          ASL (SP)        ;; CHAR OVER TO MAKE
4103 022130 006316          ASL (SP)        ;; ROOM FOR NEW ONE.
4104 022132 005266 000002 17$:  INC 2(SP)       ;; KEEP COUNT OF CHAR
4105 022136 056616 177776      BIS -2(SP),(SP) ;; SET IN NEW CHAR
4106 022142 000667          BR 7$           ;; GET THE NEXT ONE
4107 022144 104401 001214 18$:  TYPE ,SQUES     ;; TYPE ?<CR><LF>
4108 022150 000720          BR 20$          ;; SIMULATE CONTROL-U
4109 .DSABL LSB

```

```

4110
4111
4112 ;; *****
4113 ;; THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4114 ;; CALL:
4115 ;; RDCHR
4116 ;; RETURN HERE
4117 ;; GET A CHARACTER FROM THE QUEUE
4118 ;; CHARACTER IS ON THE STACK
4119 ;; WITH PARITY BIT STRIPPED OFF
4120 SRDCHR: MOV (SP),-(SP) ;; PUSH DOWN THE PC AND
4121 022154 016666 000004 000002  MOV 4(SP),2(SP) ;; THE PS
4122 022162 005066 000004      CLR 4(SP)       ;; GET READY FOR A CHARACTER
4123 022166 005046          CLR -(SP)       ;; PUT NEW PS ON STACK
4124 022170 012746 022176      MOV #64$,-(SP) ;; PUT NEW PC ON STACK
4125 022174 000002          RTI                    ;; POP NEW PC AND PS
4126 022176          64$:
4127 022176 005737 021312 1$:  TST $TKCNT     ;; WAIT ON A CHARACTER
4128 022202 001775          BEQ 1$         ;;
4129 022204 005337 021312      DEC $TKCNT     ;; DECREMENT THE COUNTER
4130 022210 117766 177102 000004  MOVB @STKQOUT,4(SP) ;; GET ONE CHARACTER

```



```

4131 022216 005237 021316          INC      STKQOUT      ;; UPDATE THE POINTER
4132 022222 023727 021316 021322  CMP      STKQOUT,#STKQEND ;; DID IT GO OFF OF THE END?
4133 022230 001003          BNE      25        ;; BRANCH IF NO
4134 022232 012737 021320 021316  MOV      #STKQRT,STKQOUT ;; RESET THE POINTER
4135 022240 000002          RTI          ;; RETURN
4136
4137 ;; *****
4138 ;; THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4139 ;; CALL:
4140 ;;
4141 ;;       RDLIN          ;; INPUT A STRING FROM THE TTY
4142 ;;       RETURN HERE   ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4143 ;;                   ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
4143 022242 010346          SRDLIN: MOV      R3,-(SP)      ;; SAVE R3
4144 022244 005046          CLR      -(SP)        ;; CLEAR THE RUBOUT KEY
4145 022246 012703 022476          15:    MOV      #STTYIN,R3      ;; GET ADDRESS
4146 022252 022703 022522          25:    CMP      #STTYIN+20.,R3  ;; BUFFER FULL?
4147 022256 101456          BLOS     45          ;; BR IF YES
4148 022260 104410          ROCHR   ;; GO READ ONE CHARACTER FROM THE TTY
4149 022262 112613          MOVB    (SP)+,(R3)    ;; GET CHARACTER
4150 022264 122713 000177          105:   CMPB    #177,(R3)    ;; IS IT A RUBOUT
4151 022270 001022          BNE     55          ;; BR IF NO
4152 022272 005716          TST     (SP)        ;; IS THIS THE FIRST RUBOUT?
4153 022274 001007          BNE     65          ;; BR IF NO
4154 022276 112737 000134 022474          MOVB    #' \ ,95     ;; TYPE A BACK SLASH
4155 022304 104401 022474          TYPE   95
4156 022310 012716 177777          MOV     #-1,(SP)    ;; SET THE RUBOUT KEY
4157 022314 005303          65:    DEC     R3        ;; BACKUP BY ONE
4158 022316 020327 022476          CMP     R3,#STTYIN  ;; STACK EMPTY?
4159 022322 103434          BLO     45          ;; BR IF YES
4160 022324 111337 022474          MOVB    (R3),95     ;; SETUP TO TYPEOUT THE DELETED CHAR.
4161 022330 104401 022474          TYPE   95          ;; GO TYPE
4162 022334 000746          BR     25          ;; GO READ ANOTHER CHR.
4163 022336 005716          55:    TST     (SP)        ;; RUBOUT KEY SET?
4164 022340 001406          BEQ     75          ;; BR IF NO
4165 022342 112737 000134 022474          MOVB    #' \ ,95     ;; TYPE A BACK SLASH
4166 022350 104401 022474          TYPE   95
4167 022354 005016          CLR     (SP)        ;; CLEAR THE RUBOUT KEY
4168 022356 122713 000025          75:    CMPB    #25,(R3)    ;; IS CHARACTER A CTRL U?
4169 022362 001003          BNE     85          ;; BR IF NO
4170 022364 104401 022527          TYPE   ,SCNTLU     ;; TYPE A CONTROL "U"
4171 022370 000726          BR     15          ;; GO START OVER
4172 022372 122713 000022          85:    CMPB    #22,(R3)    ;; IS CHARACTER A "r"?
4173 022376 001011          BNE     35          ;; BRANCH IF NO
4174 022400 105013          CLRB   (R3)        ;; CLEAR THE CHARACTER
4175 022402 104401 001215          TYPE   ,SCRLF     ;; TYPE A "CR" & "LF"
4176 022406 104401 022476          TYPE   ,STTYIN    ;; TYPE THE INPUT STRING
4177 022412 000717          BR     25          ;; GO PICKUP ANOTHER CHARACTER
4178 022414 104401 001214          45:    TYPE   ,SQUES     ;; TYPE A ' '
4179 022420 000712          BR     15          ;; CLEAR THE BUFFER AND LOOP
4180 022422 111337 022474          35:    MOVB    (R3),95     ;; ECHO THE CHARACTER
4181 022426 104401 022474          TYPE   95
4182 022432 122723 000015          CMPB    #15,(R3)+   ;; CHECK FOR RETURN
4183 022436 001305          BNE     25          ;; LOOP IF NOT RETURN
4184 022440 105063 177777          CLRB   -(R3)       ;; CLEAR RETURN (THE 15)
4185 022444 104401 001216          TYPE   ,SLF        ;; TYPE A LINE FEED
4186 022450 005726          TST     (SP)+      ;; CLEAN RUBOUT KEY FROM THE STACK

```



```

4187 022452 012603      MOV      (SP)+,R3      ;;RESTORE R3
4188 022454 011646      MOV      (SP)-(SP)    ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4189 022456 016666 000004 000002      MOV      4(SP),2(SP)  ;;FIRST ASCII CHARACTER ON IT
4190 022464 012766 022476 000004      MOV      @STTYIN,4(SP)
4191 022472 000002      RTI                ;;RETURN
4192 022474 000      9S: .BYTE 0        ;;STORAGE FOR ASCII CHAR. TO TYPE
4193 022475 000      .BYTE 0        ;;TERMINATOR
4194 022476 000024      STTYIN: .BLKB 20.  ;;RESERVE 20 BYTES FOR TTY INPUT
4195 022522 041536 005015 000      SCNTLC: .ASCIZ /?C/<15><12>  ;;CONTROL "C"
4196 022527 136 006525 000012      SCNTLU: .ASCIZ /?U/<15><12>  ;;CONTROL "U"
4197 022534 043536 005015 000      SCNTLG: .ASCIZ /?G/<15><12>  ;;CONTROL "G"
4198 022541 015 051412 051127      SMSMR: .ASCIZ <15><12>/SMR = /
4199 022546 036440 000040
4200 022552 020040 042516 020127      SMNEW: .ASCIZ / NEW = /
4201 022560 020075 000
4202 022564      .EVEN
4203
4204      .SBTTL SCOPE HANDLER ROUTINE
4205
4206      ;;*****
4207      ;;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4208      ;;AND LOAD THE TEST NUMBER($TSTNR) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4209      ;;AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
4210      ;;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4211      ;;SW14=1 LOOP ON TEST
4212      ;;SW11=1 INHIBIT ITERATIONS
4213      ;;SW09=1 LOOP ON ERROR
4214      ;;CALL
4215      ;; SCOPE ;;SCOPE=IOT
4216
4217      $SCOPE:
4218 022564 104407      CKSWR
4219 022566 032777 040000 156344 1S: BIT @BIT14,@SWR  ;;TEST FOR CHANGE IN SOFT-SWR
4220 022574 001076      BNE $OVER  ;;LOOP ON PRESENT TEST?
4221      ;;###START OF CODE FOR THE XOR TESTER###
4222 022576 000416      $XTSTR: BR 6S  ;;YES IF SW14=1
4223      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
4224 022600 013746 000004      MOV @ERRVEC,-(SP)  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
4225 022604 012737 022624 000004      MOV @S,@ERRVEC  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4226 022612 005737 177060      TST @177060  ;;SET FOR TIMEOUT
4227 022616 012637 000004      MOV (SP)+,@ERRVEC  ;;TIME OUT ON XOR?
4228 022622 000450      BR $SVLAD  ;;RESTORE THE ERROR VECTOR
4229 022624 022626      5S: CMP (SP)+,(SP)+  ;;GO TO THE NEXT TEST
4230 022626 012637 000004      MOV (SP)+,@ERRVEC  ;;CLEAR THE STACK AFTER A TIME OUT
4231 022632 000413      BR 7S  ;;RESTORE THE ERROR VECTOR
4232 022634      6S: ;;###END OF CODE FOR THE XOR TESTER###  ;;LOOP ON THE PRESENT TEST
4233 022634 105737 001103      2S: TSTB SERFLG  ;;HAS AN ERROR OCCURRED?
4234 022640 001421      BEQ 3S  ;;BR IF NO
4235 022642 123737 001115 001103      CMPEB SERMAX,SERFLG  ;;MAX. ERRORS FOR THIS TEST OCCURRED?
4236 022650 101015      SHI 3S  ;;BR IF NO
4237 022652 032777 001000 156260      BIT @BIT09,@SWR  ;;LOOP ON ERROR?
4238 022660 001404      BEQ 4S  ;;BR IF NO
4239 022662 013737 001110 001106 7S: MOV $LPERR,$LPAOR  ;;SET LOOP ADDRESS TO LAST SCOPE
4240 022670 000440      BR $OVER
4241 022672 105037 001103      4S: CLRB SERFLG  ;;ZERO THE ERROR FLAG
4242 022676 005037 001204      CLR $TIMES  ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE

```



```

4243 022702 000412          BR      15          ;; ESCAPE TO THE NEXT TEST
4244 022704 032777 004000 156226 3S:  BIT      @BIT11, @SWR  ;; INHIBIT ITERATIONS?
4245 022712 001006          BNE     15          ;; BR IF YES
4246 022714 005237 001104          INC     $ICNT      ;; INCREMENT ITERATION COUNT
4247 022720 023737 001204 001104          CMP     $TIMES, $ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
4248 022726 002021          BGE     $OVER     ;; BR IF MORE ITERATION REQUIRED
4249 022730 012737 000001 001104 1S:  MOV     @1, $ICNT  ;; REINITIALIZE THE ITERATION COUNTER
4250 022736 013737 023006 001204          MOV     $SMXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
4251 022744 105237 001102          SSVLAD: INCB    $STNM  ;; COUNT TEST NUMBERS
4252 022750 011637 001106          MOV     (SP), $LPADR  ;; SAVE SCOPE LOOP ADDRESS
4253 022754 011637 001110          MOV     (SP), $LPERR  ;; SAVE ERROR LOOP ADDRESS
4254 022760 005037 001206          CLR     $ESCAPE     ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
4255 022764 112737 000001 001115          MOV     @1, $ERMAX   ;; ONLY ALLOW ONE(I) ERROR ON NEXT TEST
4256 022772 013777 001102 156142 $OVER: MOV     $STNM, @DISPLAY ;; DISPLAY TEST NUMBER
4257 023000 013716 001106          MOV     $LPADR, (SP) ;; FUDGE RETURN ADDRESS
4258 023004 000002          RTI          ;; FIXES PS
4259 023006 000001          SMXCNT: 1      ;; MAX. NUMBER OF ITERATIONS

```

.SBTTL SAVE AND RESTORE RO-R5 ROUTINES

```

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

```

```

SSAVREG:
MOV     R0, -(SP)      ;; PUSH R0 ON STACK
MOV     R1, -(SP)      ;; PUSH R1 ON STACK
MOV     R2, -(SP)      ;; PUSH R2 ON STACK
MOV     R3, -(SP)      ;; PUSH R3 ON STACK
MOV     R4, -(SP)      ;; PUSH R4 ON STACK
MOV     R5, -(SP)      ;; PUSH R5 ON STACK
MOV     22(SP), -(SP)  ;; SAVE PS OF MAIN FLOW
MOV     22(SP), -(SP)  ;; SAVE PC OF MAIN FLOW
MOV     22(SP), -(SP)  ;; SAVE PS OF CALL
MOV     22(SP), -(SP)  ;; SAVE PC OF CALL
RTI

```

```

;#RESTORE RO-R5
;#CALL:
;# RESREG
;#RESREG:
MOV     (SP)+, 22(SP)  ;; RESTORE PC OF CALL
MOV     (SP)+, 22(SP)  ;; RESTORE PS OF CALL
MOV     (SP)+, 22(SP)  ;; RESTORE PC OF MAIN FLOW
MOV     (SP)+, 22(SP)  ;; RESTORE PS OF MAIN FLOW

```

```

4278 023010          010046
4279 023010 010046
4280 023012 010146
4281 023014 010246
4282 023016 010346
4283 023020 010446
4284 023022 010546
4285 023024 016646 000022
4286 023030 016646 000022
4287 023034 016646 000022
4288 023040 016646 000022
4289 023044 000002
4290
4291
4292
4293
4294 023046          012666 000022
4295 023046 012666 000022
4296 023052 012666 000022
4297 023056 012666 000022
4298 023062 012666 000022

```



```

4299 023066 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
4300 023070 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
4301 023072 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
4302 023074 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
4303 023076 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
4304 023100 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
4305 023102 000002      RTI

```

.SBTTL TRAP DECODER

```

;*****
;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
;GO TO THAT ROUTINE.

```

```

4315 023104 010046      STRAP:  MOV      RO,-(SP)      ;;SAVE RO
4316 023106 016600 000002  MOV      2(SP),RO          ;;GET TRAP ADDRESS
4317 023112 005740      TST      -(RO)             ;;BACKUP BY 2
4318 023114 111000      MOV      (RO),RO           ;;GET RIGHT BYTE OF TRAP
4319 023116 006300      ASL      RO                ;;POSITION FOR INDEXING
4320 023120 016000 023140  MOV      STRPAD(RO),RO      ;;INDEX TO TABLE
4321 023124 000200      RTS      RO                ;;GO TO ROUTINE

```

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

```

4326 023126 011646      STRAP2: MOV      (SP),-(SP)    ;;MOVE THE PC DOWN
4327 023130 016666 000004 000002  MOV      4(SP),2(SP)        ;;MOVE THE PSM DOWN
4328 023136 000002      RTI                        ;;RESTORE THE PSM

```

.SBTTL TRAP TABLE

```

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

```

	ROUTINE		
4337	STRPAD:	WORD	STRAP2
4338		STYPER	;;CALL=TYPER TRAP+1(104401) TTY TYPEOUT ROUTINE
4339		STYPOC	;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4340		STYPOS	;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
4341		STYPON	;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
4342		STYPDS	;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
4344		SGTSMR	;;CALL=GTSMR TRAP+6(104406) GET SOFT-SMR SETTING
4346		SCKSMR	;;CALL=CKSMR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SMR
4347		SRDCHR	;;CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
4348		SRDLIN	;;CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
4349		SSAVREG	;;CALL=SAVREG TRAP+12(104412) SAVE RO-R5 ROUTINE
4350		SRESREG	;;CALL=RESREG TRAP+13(104413) RESTORE RO-R5 ROUTINE

.SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

;*****

4351
4352
4353
4354


```

4355      ;:THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
4356      ;:UNSIGNED DECIMAL ASCIZ NUMBER.
4357      ;:CALL
4358      ;:      MOV      NUMBER, -(SP)      ;:PUT BINARY NUMBER ON THE STACK
4359      ;:      JSR      PC, @SSB2D      ;:CALL
4360      ;:      RETURN      ;:ADDRESS OF THE 1ST ASCIZ CHAR. IS ON THE STACK
4361
4362
4363      023170  016637  000002  023220  SSB2D:  MOV      2(SP), 1$      ;:SAVE BINARY NUMBER
4364      023176  012746  023220      MOV      @1$, -(SP)      ;:SET POINTER
4365      023202  004737  023224      JSR      PC, @SDB2D      ;:CALL DOUBLE LENGTH CONVERT
4366      023206  062716  000005      ADD      #5, (SP)      ;:ONLY ALLOW FIVE CHARACTERS
4367      023212  012666  000002      MOV      (SP)+, 2(SP)      ;:PICKUP POINTER
4368      023216  000207      RTS      PC      ;:RETURN
4369      023220  000000  000000      1$:      .WORD      0,0
4370
4371      .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
4372
4373      ;:*****
4374      ;:THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
4375      ;:DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
4376      ;:POSITIVE.
4377      ;:CALL
4378      ;:      MOV      #PNTR, -(SP)      ;:POINTER TO LOW WORD OF BINARY NUMBER
4379      ;:      JSR      PC, @SDB2D      ;:
4380      ;:      RETURN      ;:THE FIRST ADDRESS OF ASCIZ
4381      ;:      ;:IS ON THE STACK
4382
4383
4384      023224  104412      SDB2D:  SAVREG      ;:SAVE REGISTERS
4385      023226  016602  000002      MOV      2(SP), R2      ;:PICKUP THE DATA POINTER
4386      023232  012700  023404      MOV      #SDECVL, R0      ;:GET ADDRESS OF "SDECVL" STRING
4387      023236  010066  000002      MOV      R0, 2(SP)      ;:PUT ADDRESS OF ASCIZ STRING ON STACK
4388      023242  012201      MOV      (R2)+, R1      ;:PICKUP THE BINARY NUMBER
4389      023244  012202      MOV      (R2)+, R2
4390      023246  012737  000012  023322      MOV      #10, 4$      ;:SET UP TO DO 10 CONVERSIONS
4391      023254  012704  023334      MOV      #STNPNR, R4      ;:ADDRESS OF TEN POWER
4392      023260  012705  023336      MOV      #STNPNR+2, R5
4393      023264  005003      1$:      CLR      R3      ;:CLEAR PARTIAL
4394      023266  161401      2$:      SUB      (R4), R1      ;:SUBTRACT TEN POWER
4395      023270  005602      SBC      R2
4396      023272  161502      SUB      (R5), R2
4397      023274  002402      BLT      3$      ;:BR IF TEN POWER TO LARGE
4398      023276  005203      INC      R3      ;:ADD 1 TO PARTIAL
4399      023300  000772      BR      2$      ;:LOOP
4400      023302  062401      3$:      ADD      (R4)+, R1      ;:RESTORE SUBTRACTED VALUE
4401      023304  005502      ADC      R2
4402      023306  062402      ADD      (R4)+, R2
4403      023310  022525      CMP      (R5)+, (R5)+      ;:MOVE TO NEXT TEN POWER
4404      023312  052703  000060      BIS      #'0, R3      ;:CHANGE PARTIAL TO ASCII
4405      023316  110320      MOVB     R3, (R0)+      ;:SAVE IT
4406      023320  005327      DEC      (PC)+      ;:DONE?
4407      023322  000000      4$:      .WORD      0
4408      023324  001357      BNE     1$      ;:BR IF NO
4409      023326  105020      CLRB     (R0)+      ;:TERMINATOR
4410      023330  104413      RESREG      ;:RESTORE REGISTERS
    
```


4411 023332 000207
4412 023334 145000
4413 023336 035632
4414 023340 160400
4415 023342 002765
4416 023344 113200
4417 023346 000230
4418 023350 041100
4419 023352 000017
4420 023354 103240
4421 023356 000001
4422 023360 023420
4423 023362 000000
4424 023364 001750
4425 023366 000000
4426 023370 000144
4427 023372 000000
4428 023374 000012
4429 023376 000000
4430 023400 000001
4431 023402 000000
4432 023404 000014

RTS PC ;; RETURN
STNPR: 145000 ;; 1.0E09
35632
160400 ;; 1.0E08
2765
113200 ;; 1.0E07
230
041100 ;; 1.0E06
17
103240 ;; 1.0E05
1
23420 ;; 1.0E04
0
1750 ;; 1.0E03
0
144 ;; 1.0E02
0
12 ;; 1.0E01
0
1
0 ;; 1.0E00
SDECVL: .BLKB 12. ;; RESERVE STORAGE FOR ASCIZ STRING

.SBTTL TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
*LEADING NUMBERS.
*CALL

* MOV #NUMADR, -(SP) ;; FIRST ADDRESS OF ASCIZ STRING
* JSR PC, @SSUPRS

4444 023420 010046
4445 023422 016600 000004
4446 023426 105710
4447 023430 001403
4448 023432 122720 000060
4449 023436 001773
4450 023440 005300
4451 023442 010337 023450
4452 023446 104401
4453 023450 000000
4454 023452 012600
4455 023454 012616
4456 023456 000207

SSUPRS: MOV RO, -(SP) ;; SAVE RO
MOV 4(SP), RO ;; PICKUP THE POINTER
1S: TSTB (RO) ;; TERMINATEOR?
BEQ 2S ;; BR IF YES
CMPB #'0, (RO)+ ;; IS THIS AN ASCII "0" ?
BEQ 1S ;; BR IF YES
2S: DEC RO ;; BACKUP BY "1"
MOV RO, 3S ;; SAVE FOR TYPING
TYPE ;; GO TYPE
3S: .WORD 0 ;; ASCIZ POINTER GOES HERE
MOV (SP)+, RO ;; RESTORE RO
MOV (SP)+, (SP) ;; RESTORE THE STACK
RTS PC ;; RETURN

.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, SRAND ;; CALL THE ROUTINE
* RETURN ;; RETURN HERE THE RANDOM
* ;; NUMBER WILL BE IN

4457
4458
4459
4460
4461
4462
4463
4464
4465
4466

4467
4468
4469 023460
4470 023460 010046
4471 023462 010146
4472 023464 010246
4473 023466 013700 023560
4474 023472 013701 023556
4475 023476 012702 177771
4476 023502 006300
4477 023504 006101
4478 023506 005202
4479 023510 001374
4480 023512 063700 023560
4481 023516 005501
4482 023520 063701 023556
4483 023524 062700 001057
4484 023530 005501
4485 023532 062701 047401
4486 023536 010037 023560
4487 023542 010137 023556
4488 023546 012602
4489 023550 012601
4490 023552 012600
4491 023554 000207
4492 023556 176543
4493 023560 123456

```
;*
SRAND:
MOV RO,-(SP) ;; PUSH RO ON STACK
MOV R1,-(SP) ;; PUSH R1 ON STACK
MOV R2,-(SP) ;; PUSH R2 ON STACK
MOV SLONUM,RO ;; SET RO WITH LOW
MOV SHINUM,R1 ;; SET R1 WITH HIGH
MOV #7,R2 ;; SET SHIFT COUNT
1S: ASL RO ;; SHIFT RO LEFT AND
ROL R1 ;; ROTATE CARRY INTO R1 AND
INC R2 ;; CHECK FOR DONE
BNE 1S ;; CONTINUE SHIFT LOOP
ADD SLONUM,RO ;; ADD NUMBER TO MAKE X 129
ADC R1 ;; PROPOGATE CARRY
ADD SHINUM,R1 ;; ADD NUMBER TO MAKE X 129
ADD #1057,RO ;; ADD LOW CONSTANT
ADC R1 ;; PROPOGATE CARRY
ADD #47401,R1 ;; ADD HIGH CONSTANT
MOV RO,SLONUM ;; SAVE RO
MOV R1,SHINUM ;; SAVE R1
MOV (SP)+,R2 ;; POP STACK INTO R2
MOV (SP)+,R1 ;; POP STACK INTO R1
MOV (SP)+,RO ;; POP STACK INTO RO
RTS PC ;; RETURN
SHINUM: .WORD 176543
SLONUM: .WORD 123456
```

.SBTTL INTEGER DIVIDE ROUTINE

4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520 023562
4521 023562 104400
4522 023564 042716 000017

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

STACK NO ERROR OVERFLOW DIVIDE BY ZERO
-----
TOP REMAINDER ALL ZEROS ALL ONES
+2 QUOTIENT ALL ZEROS ALL ONES

SDIV:
TRAP
BIC #17,(SP) ;; PUSH OLD PSH AND PC ON STACK
;; STRIP AWAY CONDITION CODES
```


4523	023570	010046		MOV	RO,-(SP)	::PUSH RO ON STACK	
4524	023572	010146		MOV	R1,-(SP)	::PUSH R1 ON STACK	
4525	023574	010246		MOV	R2,-(SP)	::PUSH R2 ON STACK	
4526	023576	010346		MOV	R3,-(SP)	::PUSH R3 ON STACK	
4527	023600	005046		CLR	-(SP)	::SAVE A PLACE FOR SIGNS	
4528	023602	012746	000021	MOV	#17,-(SP)	::SETUP THE ITERATION COUNTER	
4529	023606	016601	000024	MOV	24(SP),R1	::PICKUP THE DIVIDEND	
4530	023612	016600	000022	MOV	22(SP),RO		
4531	023616	100005		BPL	1\$::CHECK THE SIGN	
4532	023620	105366	000003	DECB	3(SP)	::KEEP TRACK OF THE SIGN	
4533	023624	005400		NEG	RO	::AND NEGATE THE ORIGINAL	
4534	023626	005401		NEG	R1	::NUMBER	
4535	023630	005600		SBC	RO		
4536	023632	016602	000020	15:	MOV	20(SP),R2	::PICKUP THE DIVISOR
4537	023636	002407		BLT	2\$::CHECK THE SIGN	
4538	023640	003011		BGT	3\$::DIVISOR OF 0 IS A NO-NO	
4539	023642	052766	000003 000014	BIS	#3,14(SP)	::SET "V" & "C"	
4540	023650	012700	177777	MOV	#-1,RO	::SET REMAINDER TO ALL ONES	
4541	023654	000424		BR	7\$::EXIT	
4542	023656	005266	000002	25:	INC	2(SP)	::KEEP TRACK OF DIVISORS SIGN
4543	023662	000401		BR	4\$		
4544	023664	005402		35:	NEG	R2	::NEGATE THE ORIGINAL NUMBER
4545	023666	000241		45:	CLC		::CLEAR "C"
4546	023670	000405		BR	6\$::START FORMING QUOTIENT
4547	023672	006100		55:	ROL	RO	::POSITION MSB'S
4548	023674	010003		MOV	RO,R3	::COPY	
4549	023676	060203		ADD	R2,R3	::COMPARE DIVIDEND & DIVISOR	
4550	023700	103001		BCC	6\$::BR IF DIVIDEND > DIVISOR	
4551	023702	010300		MOV	R3,RO	::REMAINDER AFTER THIS LOOP	
4552	023704	006101		65:	ROL	R1	::QUOTIENT BIT ENTERS HERE
4553	023706	005316		DEC	(SP)	::DONE?	
4554	023710	001370		BNE	5\$::BR IF NO	
4555	023712	005701		TST	R1	::OVERFLOW?	
4556	023714	100005		BPL	8\$::BR IF NO	
4557	023716	052766	000002 000014	BIS	#2,14(SP)	::SET "V" IN RETURN STATUS WORD	
4558	023724	005000		CLR	RO	::SET REMAINDER TO ALL ZEROS	
4559	023726	010001		75:	MOV	RO,R1	::COPY REMAINDER INTO QUOTIENT
4560	023730	005726		85:	TST	(SP)+	::CLEAR COUNTER FROM STACK
4561	023732	005716		TST	(SP)	::REMAINDER SIGN CORRECTION NEEDED?	
4562	023734	002004		BGE	9\$::BR IF NO	
4563	023736	005400		NEG	RO	::NEGATE REMAINDER	
4564	023740	105066	000001	CLRB	1(SP)	::CLEAR SIGN	
4565	023744	005316		DEC	(SP)	::BUT DON'T FORGET QUOTIENT	
4566	023746	005726		95:	TST	(SP)+	::QUOTIENT SIGN CORRECTION NEEDED?
4567	023750	001401		BEQ	10\$::BR IF NO	
4568	023752	005401		NEG	R1	::NEGATE QUOTIENT	
4569	023754	010166	000020	105:	MOV	R1,20(SP)	::RETURN QUOTIENT AND
4570	023760	010066	000016	MOV	RO,16(SP)	::REMAINDER TO USER	
4571	023764	012603		MOV	(SP)+,R3	::POP STACK INTO R3	
4572	023766	012602		MOV	(SP)+,R2	::POP STACK INTO R2	
4573	023770	012601		MOV	(SP)+,R1	::POP STACK INTO R1	
4574	023772	012600		MOV	(SP)+,RO	::POP STACK INTO RO	
4575	023774	012666	000002	MOV	(SP)+,2(SP)	::SETUP TO RETURN CONDITION CODES	
4576	024000	000002		RTI		::RETURN	
4577							
4578							

.SBTTL *** PROGRAM SUBROUTINES ***


```

4579
4580
4581      ;SET "LPTAVL" TO THE PROPER STATE.
4582      LPTAVL = 0 IF NO LINE PRINTER AVAILABLE
4583      LPTAVL = 1 IF LINE PRINTER IS AVAILABLE
4584      CALL
4585      JSR      PC, @#LP.AVL
4586      RETURN
4587 024002 005037 001230 LP.AVL: CLR      @#LPTAVL      ;START WITH NO PRINTER AVAIABLE
4588 024006 012737 024032 000004 MOV      @#ERRVEC      ;SETUP THE TIMEOUT VECTOR
4589 024014 005037 000006 CLR      @#ERRVEC+2
4590 024020 005777 155374 TST      @#LPS          ;IS THERE A LINE PRINTER?
4591 024024 005237 001230 INC      @#LPTAVL      ;YES--SET AVAILABLE SWITCH
4592 024030 000401 BR      2$
4593 024032 022626 1$:      CMP      (SP)+, (SP)+      ;NO--POP STACK
4594 024034 012737 000006 000004 2$:      MOV      @#ERRVEC+2, @#ERRVEC ;RESTORE TIMEOUT VECTOR
4595 024042 000207 RTS      PC              ;RETURN
4596
4597      ;THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
4598      ;AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK
4599      ;"CLKSTA" WILL INDICATE THE CLOCK TYPE
4600      0= NO CLOCK
4601      +1= KW11-P
4602      -1= KW11-L
4603      ;THIS ROUTINE WILL ALSO SETUP "TICKMS" (TIME
4604      ;PER CLOCK TICK IN MILLISECONDS) AND "TICKUS"
4605      ;(TIME PER CLOCK TICK IN MICROSECONDS) AS
4606      ;PER SW00.
4607      SW00=0 -- 60HZ
4608      SW00=1 -- 50HZ
4609      CALL
4610      JSR      PC, @#ST.CLK
4611      RETURN
4612
4613 024044 010146 ST.CLK: MOV      R1, -(SP)      ;SAVE R1
4614 024046 012701 000006 MOV      @#ERRVEC+2, R1 ;SAVE AND SETUP TIMEOUT VECTOR
4615 024052 011146 MOV      (R1), -(SP)
4616 024054 005011 CLR      (R1)          ;LEVEL 0
4617 024056 014146 MOV      -(R1), -(SP)
4618 024060 012711 024110 MOV      @#1$, (R1)      ;GO TO 1$ ON TIMEOUT
4619 024064 005037 001244 CLR      CLKSTA        ;SET CLOCK STATUS TO NO CLOCK
4620 024070 005777 155304 TST      @#PKCS        ;IS THERE A KW11-P?
4621 024074 012737 000001 001244 MOV      @#1, CLKSTA    ;YES--SET STATUS TO KW11-P
4622 024102 004737 024212 JSR      PC, ST.PCLK    ;START THE KW11-P
4623 024106 000414 BR      3$            ;GO TO EXIT
4624 024110 022626 1$:      CMP      (SP)+, (SP)+      ;CLEAN UP THE STACK
4625 024112 012711 024136 MOV      @#2$, (R1)      ;IF TIMEOUT GO TO 2$
4626 024116 005777 155270 TST      @#LKS          ;IS THERE A KW11-L?
4627 024122 012737 177777 001244 MOV      @#-1, CLKSTA   ;YES-- SET STATUS TO KW11-L
4628 024130 004737 024254 JSR      PC, ST.LCLK    ;START THE KW11-L
4629 024134 000401 BR      3$            ;EXIT
4630 024136 022626 2$:      CMP      (SP)+, (SP)+      ;CLEAN UP THE STACK
4631 024140 012621 3$:      MOV      (SP)+, (R1)+      ;RESTORE THE TIMEOUT VECTOR
4632 024142 012621 MOV      (SP)+, (R1)+
4633 024144 012601 MOV      (SP)+, R1      ;RESTORE R1
4634 024146 032737 000100 001220 BIT      @#SW06, @#C.SWR ;50HZ OR 60HZ?
    
```



```

4635 024154 001407          BEQ      4$          ;BRANCH IF 60
4636 024156 012737 000020 001246  MOV     #20,#TICKMS ;SETUP TIME PER
4637 024164 012737 047040 001250  MOV     #20000.,#TICKUS ;TICK FOR 50HZ
4638 024172 000406          BR       5$
4639 024174 012737 000016 001246 4$:  MOV     #16,#TICKMS ;SETUP TIME PER
4640 024202 012737 040432 001250  MOV     #16666.,#TICKUS ;TICK FOR 60HZ
4641 024210 000207          5$:  RTS      PC          ;RETURN
4642
4643 024212          ST.PCLK:
4644 024212 032737 000040 001220  BIT     #SMOS,#C.SWR ;ALLOW SOFTWARE TIMEOUTS?
4645 024220 001014          BNE     1$          ;NO--BRANCH
4646 024222 012777 024310 155144  MOV     #SRVCLK,#PKV ;SETUP THE KW11-P VECTOR
4647 024230 012777 000300 155140  MOV     #300,#PKV+2
4648 024236 012777 000001 155136  MOV     #1,#PKB      ;COUNT ONE TICK
4649 024244 012777 000115 155126  MOV     #115,#PKCS  ;"INT.EN.",COUNT DOWN,"MODE 1 (REPEAT)",
4650                                     ;"LINE FREQ",AND "RUN"
4651 024252 000207          1$:  RTS      PC          ;RETURN
4652
4653 024254          ST.LCLK:
4654 024254 032737 000040 001220  BIT     #SMOS,#C.SWR ;ALLOW SOFTWARE TIMEOUTS?
4655 024262 001011          BNE     1$          ;NO--BRANCH
4656 024264 012777 024310 155114  MOV     #SRVCLK,#LKV ;SETUP THE KW11-L VECTOR
4657 024272 012777 000300 155110  MOV     #300,#LKV+2
4658 024300 012777 000100 155104  MOV     #100,#LKS   ;START THE KW11-L
4659 024306 000207          1$:  RTS      PC          ;RETURN
4660
4661 024310 013746 001246  SRVCLK: MOV     #TICKMS,-(SP) ;TIME PER TICK IN MILLISECONDS
4662 024314 004737 040704  JSR     PC,#RPTMR   ;COUNT THE ELAPSED TIME
4663 024320 000002          RTI          ;RETURN AFTER INTERRUPT
4664
4665                                     ;THIS ROUTINE SETS UP DEFAULT PARAMETER VALUES WHEN THE PROGRAM IS
4666                                     ;STARTED OR WHEN THE VALUE OF BIT00 IN 'C.SWR' IS CHANGED.
4667                                     ;CALL
4668                                     ; JSR     PC,LODFLT
4669                                     ;
4670
4671 024322          LODFLT:
4672 024322 010046          MOV     R0,-(SP)    ;:PUSH R0 ON STACK
4673 024324 010146          MOV     R1,-(SP)    ;:PUSH R1 ON STACK
4674 024326 010246          MOV     R2,-(SP)    ;:PUSH R2 ON STACK
4675 024330 010346          MOV     R3,-(SP)    ;:PUSH R3 ON STACK
4676 024332 012737 176777 001234  MOV     #176777,TSTNMS ;:SELECT TESTS 0-10,12-17
4677 024340 012737 000001 001236  MOV     #1,TSTNMS+2 ;:SET SELECT BIT FOR TEST 20
4678 024346 012700 001664          MOV     #DFLT,R0    ;:DEFAULT PARAMETERS POINTER
4679 024352 012701 002330          MOV     #PRMO,R1    ;:TABLE POINTER
4680 024356 010102          MOV     R1,R2      ;:STOP ADDRESS
4681 024360 012021          1$:  MOV     (R0)+,(R1)+ ;:MOVE DEFAULT PARAMETERS INTO
4682 024362 020002          CMP     R0,R2      ;:RUN TIME TABLES ** DONE?
4683 024364 103775          BLO     1$          ;:NO--BRANCH
4684 024366 012700 003504          MOV     #PAT8,R0    ;:PATO DEFAULTS TO PATTERN 8
4685 024372 012701 003104          MOV     #PATO,R1
4686 024376 012021          2$:  MOV     (R0)+,(R1)+
4687 024400 020027 003544          CMP     R0,#PAT9
4688 024404 103774          BLO     2$
4689 024406 032737 000001 001220  BIT     #BIT00,C.SWR ;:16 BIT MODE ?
4690 024414 001012          BNE     3$          ;:BR IF 18
    
```



```

4691 024416 012737 000025 001630      MOV      #21.,PRMLMT+22 ;SET 'FS' LIMIT TO 21.
4692 024424 012737 000025 001632      MOV      #21.,PRMLMT+24 ;SET 'LS' LIMIT TO 21.
4693 024432 012737 165000 001352      MOV      #- (256.*22.),TRCKWC ;WORD COUNT FOR A 16 BIT TRACK
4694 024440 000411                BR       4$ ;CONTINUE
4695 024442 012737 000023 001630 3$:      MOV      #19.,PRMLMT+22 ;SET 'FS' LIMIT TO 19.
4696 024450 012737 000023 001632      MOV      #19.,PRMLMT+24 ;SET 'LS' LIMIT TO 19.
4697 024456 012737 166000 001352      MOV      #- (256.*20.),TRCKWC ;WORD COUNT FOR COUNT FOR AN 18 BIT TRACK
4698 024464 012701 001536                MOV      #PRMPT,R1 ;ADDRESS OF PARAMETER POINTER TABLE
4699 024470 005711                TST      (R1) ;END OF THE TABLE ?
4700 024472 001425                BEQ     8$ ;BR IF END
4701 024474 032731 002000                BIT     #BIT10,0(R1)+ ;'LS' SELECTED ?
4702 024500 001773                BEQ     5$ ;BR IF NOT
4703 024502 016102 177776                MOV     -2(R1),R2 ;PARAMETER TABLE ADDRESS
4704 024506 011246                MOV     (R2),-(SP) ;PARAMETER ALLOCATION BITS
4705 024510 012703 000013                MOV     #11.,R3 ;NUMBER OF PARAMETERS (MAXIMUM) BEFORE 'LS'
4706 024514 006216                ASR     (SP) ;COUNT THE PARAMETER
4707 024516 103002                BCC     7$ ;BR IF NOT USED
4708 024520 062702 000002                ADD     #2,R2 ;INCREMENT THE PARAMETER TABLE ADDRESS
4709 024524 005303                DEC     R3 ;COUNT THE PARAMETER
4710 024526 001372                BNE     6$ ;BR IF NOT THERE YET
4711 024530 005726                TST     (SP)+ ;CORRECT THE STACK POINTER
4712 024532 021237 001630                CMP     (R2),PRMLMT+22 ;IS 'LS' TOO LARGE FOR THE MODE SELECTED ?
4713 024536 101754                BLOS   5$ ;BR IF NOT
4714 024540 013712 001630                MOV     PRMLMT+22,(R2) ;RESET VALUE FOR MODE USED
4715 024544 000751                BR      5$ ;CONTINUE
4716 024546                BS:
4717 024546 012603                MOV     (SP)+,R3 ;:POP STACK INTO R3
4718 024550 012602                MOV     (SP)+,R2 ;:POP STACK INTO R2
4719 024552 012601                MOV     (SP)+,R1 ;:POP STACK INTO R1
4720 024554 012600                MOV     (SP)+,R0 ;:POP STACK INTO R0
4721 024556 000207                RTS     PC ;RETURN
4722
4723 ;THIS ROUTINE FILLS THE PARAMETER TABLE THE CURRENT TEST.
4724 ;CALL
4725 ;
4726 ;
4727 ;
4728 ;
4729 ;
4730 024560                LODPRM:
4731 024560 010146                MOV     R1,-(SP) ;:PUSH R1 ON STACK
4732 024562 010246                MOV     R2,-(SP) ;:PUSH R2 ON STACK
4733 024564 010346                MOV     R3,-(SP) ;:PUSH R3 ON STACK
4734 024566 010446                MOV     R4,-(SP) ;:PUSH R4 ON STACK
4735 024570 005004                CLR     R4 ;CLEAR R4
4736 024572 113704 001102                MOV     $STSTNM,R4 ;GET THE TEST NUMBER
4737 024576 006304                ASL     R4 ;SETUP TO ADDRESS WORDS
4738 024600 016401 001536                MOV     PRMPT(R4),R1 ;GET THE TEST'S PARAMETER TABLE ADDRESS
4739 024604 012702 001504                MOV     #PRM,R2 ;PARAMETER EXECUTION TABLE
4740 024610 005003                CLR     R3 ;R3 IS USED AS A COUNTER
4741 024612 013704 001254                MOV     CHKDRV,R4 ;DRIVE'S ADDRESS
4742 024616 012122                MOV     (R1)+,(R2)+ ;PARAMETER SPECIFIER
4743 024620 006237 001504                ASR     PRM ;THIS PARAMETER USED IN THE TEST ?
4744 024624 103002                BCC     2$ ;BR IF NOT
4745 024626 012122                MOV     (R1)+,(R2)+ ;LOAD THE VALUE
4746 024630 000401                BR      3$ ;CONTINUE
4746 024632 005022                BS:      CLR     (R2)+ ;CLEAR THE UNUSED PARAMETER LOCATION
    
```


000

```

4747 024634 005203          3S:  INC      R3          ;COUNT THE POSITION IN THE OUTPUT TABLE
4748 024636 020327 000014    CMP      R3,#12.       ;FINISHED ?
4749 024642 001430          BEQ      6S             ;BR IF YES
4750 024644 020327 000003    CMP      R3,#3         ;DOING THE CYLINDER ADDRESSES ?
4751 024650 001363          BNE      1S             ;BR IF NOT
4752 024652 132764 000004 034346 BITB     #BIT02,DRVTP(R4) ;RPO6 ?
4753 024660 001016          BNE      5S             ;BR IF IT IS
4754 024662 062703 000002    ADD      #2,R3         ;COUNT THE BYPASSED PARAMETERS (FC' & LC')
4755 024666 006237 001504    ASR      PRM           ;SHIFT THE COUNTER
4756 024672 103002          BCC      4S             ;BR IF FC' IS NOT USED
4757 024674 062701 000002    ADD      #2,R1         ;MOVE THE INPUT POINTER
4758 024700 006237 001504    4S:  ASR      PRM           ;COUNT THE PARAMETER
4759 024704 103345          BCC      1S             ;BR IF LC' NOT USED
4760 024706 062701 000002    ADD      #2,R1         ;MOVE THE INPUT PINTER
4761 024712 000742          BR       1S             ;KEEP GOING
4762 024714 000741          BR       1S             ;KEEP GOING
4763 024716 162702 000004    5S:  SUB      #4,R2         ;BACKUP THE OUTPUT POINTER
4764 024722 000736          BR       1S             ;KEEP GOING
4765 024724          6S:
4766 024724 012604          MOV      (SP)+,R4      ;;POP STACK INTO R4
4767 024726 012603          MOV      (SP)+,R3      ;;POP STACK INTO R3
4768 024730 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
4769 024732 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
4770 024734 000207          RTS      PC            ;RETURN
  
```

;; THIS ROUTINE LOADS A READ HEADER AND DATA COMMAND OR A SEEK COMMAND
 ;; INTO DPB.B+2 AND DPB.C+2, DEPENDING ON THE STATE OF "CONTROL SWITCH"
 ;; BIT07.

```

4771          ;CALL
4772          ;
4773          ;
4774          ;
4775          ;CALL
4776          ;
4777          ;
4778          ;
4779 024736 032737 000200 001220 LDCMD:  BIT      #SW07,#C.SMR ;DO EXPLICIT SEEKS?
4780 024744 001007          BNE      1S             ;YES--BRANCH
4781 024746 012737 000173 004126    MOV      #READHD,#DPB.B+2 ;NO--SET UP FOR READ HEADER AND
4782 024754 012737 000173 004146    MOV      #READHD,#DPB.C+2 ;DATA COMMAND
4783 024762 000406          BR       2S             ;
4784 024764 012737 000105 004126    1S:  MOV      #SEEK,#DPB.B+2 ;SETUP FOR SEEK COMMAND
4785 024772 012737 000105 004146    MOV      #SEEK,#DPB.C+2
4786 025000 000207          2S:  RTS      PC
  
```

;; THIS ROUTINE WILL CALL THE RPO4/5/6 DRIVER AND THEN WAIT ON THE FUNCTION
 ;; TO COMPLETE. IF AN ERROR OCCURS IT IS REPORTED.

```

4787          ;CALL
4788          ;
4789          ;
4790          ;
4791          ;CALL
4792          ;
4793          ;
4794          ;
4795 025002 005037 001206          CALL.A: CLR      #SESCAPE ;NO ESCAPE ADDRESS
4796 025006 004037 035252          JSR      R0,#RPO4      ;CALL RPO4 DRIVER
4797 025012 004104          DPB.A
4798 025014 000772          BR       CALL.A
4799 025016 005737 004122    1S:  TST      #DPB.A+16    ;DONE?
4800 025022 001775          BEQ      1S             ;NO--LOOP
4801 025024 100032          BPL      3S             ;BRANCH IF NO ERROR
4802 025026 012737 025102 001206    MOV      #2S,SESCAPE  ;;ESCAPE TO 2S ON ERROR
  
```



```

4803 025034 013737 004116 001270      MOV      2#DPB.A+12,2#CYL.DS ;CYLINDER
4804 025042 113737 004115 001274      MOVB     2#DPB.A+11,2#TRK.DS ;TRACK
4805 025050 113737 004114 001272      MOVB     2#DPB.A+10,2#SEC.DS ;SECTOR
4806 025056 012746 004122      MOV      2#DPB.A+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
4807 025062 004737 026216      JSR      PC,2#ERINDX ;FORM DISPATCH INDEX
4808 025066 062607      ADD      (SP)+,PC ;REPORT PROPER ERROR
4809 025070 104041      ERROR   41 ;
4810 025072 104042      ERROR   42 ;PARITY ERROR
4811 025074 104043      ERROR   43 ;UNSAFE ERROR
4812 025076 104044      ERROR   44 ;NON-I/O ERROR
4813 025100 104045      ERROR   45 ;I/O ERROR
4814 025102 013746 004122 2S:  MOV      DPB.A+16,-(SP) ;STATUS WORD
4815 025106 004737 026156      JSR      PC,LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
4816 025112 000200      3S:  RTS      RD ;RETURN

```

THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND IF THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK, AND SECTOR) READ IS CHECKED FOR VALIDITY.

```

4817
4818
4819
4820
4821
4822
4823
4824
4825
4826 025114 005037 001206      CALL    CALL.B
4827 025120 004037 025252      JSR      RD,2#CALL.B
4828 025124 004124      RETURN
4829 025126 000772
4830 025130 005737 004142 1S:  TST      DPB.B+16 ;DONE?
4831 025134 001775      BEQ     1S ;NO--BRANCH
4832 025136 100042      BPL     4S ;BRANCH IF NO ERROR
4833 025140 012737 025232 001206      MOV      3S,SESCAPE ;ESCAPE TO 3S ON ERROR
4834 025146 013737 004136 001270      MOV      2#DPB.B+12,2#CYL.DS ;CYLINDER
4835 025154 113737 004135 001274      MOVB     2#DPB.B+11,2#TRK.DS ;TRACK
4836 025162 113737 004134 001272      MOVB     2#DPB.B+10,2#SEC.DS ;SECTOR
4837 025170 012746 004142      MOV      2#DPB.B+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
4838 025174 004737 026216      JSR      PC,2#ERINDX ;FORM DISPATCH INDEX
4839 025200 062607      ADD      (SP)+,PC ;REPORT PROPER ERROR
4840 025202 104041      ERROR   41 ;
4841 025204 104042      ERROR   42 ;PARITY ERROR
4842 025206 104043      ERROR   43 ;UNSAFE ERROR
4843 025210 104044      ERROR   44 ;NON-I/O ERROR
4844 025212 005737 004220      TST      RP.REG+RPER1 ;DRIVE ERROR?
4845 025216 001404      BEQ     2S ;BR IF NOT
4846 025220 032737 177677 004220      BIT      81C100,RP.REG+RPER1 ;SEE IF ONLY 'HCE' SET
4847 025226 001406      BEQ     4S ;BR IF IT IS
4848 025230 104045      2S:  ERROR   45 ;I/O ERROR
4849 025232 013746 004142 3S:  MOV      DPB.B+16,-(SP) ;STATUS WORD
4850 025236 004737 026156      JSR      PC,LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
4851 025242 000410      BR      5S ;CHECK FOR STALL
4852 025244 123727 004126 000173 4S:  CHPB    2#DPB.B+2,2#READHD ;DOING IMPLIED SEEKS?
4853 025252 001004      BNE     5S ;NO--BRANCH
4854 025254 004037 026476      JSR      RD,2#VERIFY ;YES--GO CHECK THE DATA
4855 025260 004134      DPB.B+10
4856 025262 000407      BR      6S ;ERROR DURING VERIFY
4857 025264 032737 040000 001220 5S:  BIT      8SH14,2#C.SMR ;STALL?
4858 025272 001403      BEQ     6S ;NO--BRANCH

```



```

4859 025274 004037 026334      JSR    RO, @STALL      ;YES--CALL STALL ROUTINE
4860 025300 001354              .WORD  STALL1        ;STALL TIME POINTER
4861 025302 000200      6S:   RTS    RO      ;RETURN
4862
4863      ; THIS ROUTINE IS THE SAME AS "CALL.B" EXCEPT FOR THE DPB USED.
4864      ; CALL
4865      ;     FILL DPB
4866      ;     JSR    RO, @CALL.C
4867      ;     RETURN
4868
4869 025304 005037 001206      CALL.C: CLR    @SESCAPE  ;NO ESCAPE ADDRESS
4870 025310 004037 035252      JSR    RO, @RPO4     ;CALL RPO4 DRIVER
4871 025314 004144      DPB.C
4872 025316 000772      BR     CALL.C
4873 025320 005737 004162      1S:   TST    @DPB.C+16 ;DONE?
4874 025324 001775      BEQ    1S            ;NO--LOOP
4875 025326 100042      BPL    4S            ;YES--BRANCH IF NO ERROR
4876 025330 012737 025422 001206      MOV    @3S, @SESCAPE ;ESCAPE TO 3S ON ERROR
4877 025336 013737 004156 001270      MOV    @DPB.C+12, @CYL.DS ;CYLINDER
4878 025344 113737 004155 001274      MOVB  @DPB.C+11, @TRK.DS ;TRACK
4879 025352 113737 004154 001272      MOVB  @DPB.C+10, @SEC.DS ;SECTOR
4880 025360 012746 001162      MOV    @DPB.C+16, -(SP) ;STATUS/ERROR INDICATOR ADDRESS
4881 025364 004737 026216      JSR    PC, @ERRINDX  ;FORM DISPATCH INDEX
4882 025370 062607      ADD    (SP)+, PC     ;REPORT PROPER ERROR
4883 025372 104041      ERROR  41
4884 025374 104042      ERROR  42            ;PARITY ERROR
4885 025376 104043      ERROR  43            ;UNSAFE ERROR
4886 025400 104044      ERROR  44            ;NON-I/O ERROR
4887 025402 005737 004220      TST    RP.REG+RPER1  ;DRIVE ERROR ?
4888 025406 001404      BEQ    2S            ;BR IF NOT
4889 025410 032737 177677 004220      BIT    @C100, RP.REG+RPER1 ;SEE IF ONLY 'HCE' SET
4890 025416 001406      BEQ    4S            ;BR IF IT IS
4891 025420 104045      2S:   ERROR  45            ;I/O ERROR
4892 025422 013746 004162      3S:   MOV    DPB.C+16, -(SP) ;STATUS WORD
4893 025426 004737 026156      JSR    PC, LOP.CK    ;SEE IF LOOP, ABORT, OR CONTINUE
4894 025432 000410      BR     5S
4895 025434 123727 004146 000173 4S:   CMPB  @DPB.C+2, @READHD ;DOING IMPLIED SEEK?
4896 025442 001004      BNE    5S            ;NO--EXIT
4897 025444 004037 026476      JSR    RO, @VERIFY   ;YES--CHECK THE DATA
4898 025450 004154      DPB.C+10
4899 025452 000407      BR     6S            ;ERRR DURING VERIFY
4900 025454 032737 040000 001220 5S:   BIT    @SW14, @C.SMR ;STALL?
4901 025462 001403      BEQ    6S            ;NO--BRANCH
4902 025464 004037 026334      JSR    RO, @STALL    ;YES--CALL STALL ROUTINE
4903 025470 001354              .WORD  STALL1        ;STALL TIME POINTER
4904 025472 000200      6S:   RTS    RO
4905
4906
4907      ; THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND
4908      ; ON AN ERROR LOCATION "ERR.CT" IS EXAMINED. IF ERR.CT IS EQUAL TO
4909      ; SERFLG EXIT IS TO THE NEXT TEST.
4910      ; CALL
4911      ;     FILL DPB
4912      ;     JSR    RO, @DRVCAL
4913      ;     RETURN
4914

```



```

4915 025474 005037 001206      DRVCAL: CLR      @#SESCAPE      : NO ESCAPE ADDRESS
4916 025500 005037 001334      CLR      @#WCEFLG      : CLEAR WRITE CHECK ERROR FLAG
4917 025504 004037 035252      JSR      RO,@#RPO4      : CALL RPO4 DRIVER
4918 025510 004164      DTADPB
4919 025512 000770      BR
4920 025514 005737 004202      DRVCL1: TST      @#DTADPB+16    : DONE
4921 025520 001775      BEQ      DRVCL1         : NO--LOOP
4922 025522 100402      BMI     15             : BR IF ERRORS
4923 025524 000137 026136      JMP      10S          : NO ERRORS
4924 025530
4925 025530 012737 025604 001206      15:  MOV      @2S,SESCAPE      : ESCAPE TO 2S ON ERROR
4926 025536 013737 004176 001270      MOV      @#DTADPB+12,@#CYL.DS : CYLINDER
4927 025544 113737 004175 001274      MOVB    @#DTADPB+11,@#TRK.DS  : TRACK
4928 025552 113737 004174 001272      MOVB    @#DTADPB+10,@#SEC.DS  : SECTOR
4929 025560 012746 004202      MOV      @#DTADPB+16,-(SP)    : STATUS/ERROR INDICATOR ADDRESS
4930 025564 004737 026216      JSR      PC,@#ERINDX        : FORM DISPATCH INDEX
4931 025570 062607      ADD      (SP)+,PC          : REPORT PROPER ERROR
4932 025572 104041      ERROR   41
4933 025574 104042      ERROR   42             : PARITY ERROR
4934 025576 104043      ERROR   43             : UNSAFE ERROR
4935 025600 104044      ERROR   44             : NON-I/O ERROR
4936 025602 104045      ERROR   45             : I/O ERROR
4937 025604 122737 000020 001102      25:  CHPB    @20,@#STSTNM        : TEST 20?
4938 025612 001137      BNE     8S             : NO--BRANCH
4939 025614 013746 004202      MOV      DTADPB+16,-(SP)    : STATUS WORD
4940 025620 004737 026156      JSR      PC,LOP.CK        : SEE IF LOOP ABORT, OR CONTINUE
4941 025624 122737 000151 004166      CHPB    @#WCKD,@#DTADPB+2    : DOING A WRITE CHECK?
4942 025632 001133      BNE     12S          : NO--BRANCH
4943 025634 032737 040000 004214      BIT     @BIT14,@#RP.REG+10 : IS "WCE"=1?
4944 025642 001527      BEQ     12S          : NO--BRANCH
4945 025644 032777 000020 153266      BIT     @5M04,@#SMR        : INHIBIT WRITES?
4946 025652 001123      BNE     12S          : YES--BRANCH
4947 025654 112737 000161 004166      MOVB    @WRITE,@#DTADPB+2    : SETUP FOR A WRITE
4948 025662 005037 001206      CLR     @#SESCAPE        : NO ESCAPE ADDRESS
4949 025666 004037 035252      JSR      RO,@#RPO4        : DO THE WRITE
4950 025672 004164      DTADPB
4951 025674 000240      NOP
4952 025676 005737 004202      35:  TST     @#DTADPB+16        : DONE?
4953 025702 001775      BEQ     3S             : NO--LOOP
4954 025704 100026      BPL     4S             : YES--BRANCH IF NO ERROR
4955 025706 012737 026112 001206      MOV      @8S,SESCAPE        : ESCAPE TO 8S ON ERROR
4956 025714 013737 004176 001270      MOV      @#DTADPB+12,@#CYL.DS : CYLINDER
4957 025722 113737 004175 001274      MOVB    @#DTADPB+11,@#TRK.DS  : TRACK
4958 025730 113737 004174 001272      MOVB    @#DTADPB+10,@#SEC.DS  : SECTOR
4959 025736 012746 004202      MOV      @#DTADPB+16,-(SP)    : STATUS/ERROR INDICATOR ADDRESS
4960 025742 004737 026216      JSR      PC,@#ERINDX        : FORM DISPATCH INDEX
4961 025746 062607      ADD      (SP)+,PC          : REPORT PROPER ERROR
4962 025750 104041      ERROR   41
4963 025752 104042      ERROR   42             : PARITY ERROR
4964 025754 104043      ERROR   43             : UNSAFE ERROR
4965 025756 104044      ERROR   44             : NON-I/O ERROR
4966 025760 104045      ERROR   45             : I/O ERROR
4967 025762 112737 000151 004166      45:  MOVB    @#WCKD,@#DTADPB+2    : COMMAND=WRITE CHECK DATA
4968 025770 004037 035252      JSR      RO,@#RPO4        : DO THE WRITE CHECK
4969 025774 004164      DTADPB
4970 025776 000240      NOP
    
```



```

4971 026000 005737 004202 5S: TST 2#DTADPB+16 ;DONE?
4972 026004 001775 BEQ 5S ;NO--LOOP
4973 026006 100410 BMI 7S ;YES--BRANCH IF ERROR
4974 026010 004037 035252 JSR RD,2#RPO4 ;DO A 2ND WRITE CHECK
4975 026014 004164 DTADPB
4976 026016 000240 NOP
4977 026020 005737 004202 6S: TST 2#DTADPB+16 ;DONE?
4978 026024 001775 BEQ 6S ;NO--LOOP
4979 026026 100043 BPL 10S ;YES--BRANCH IF NO ERROR
4980 026030 012737 000001 001334 7S: MOV 2#WCEFLG ;SET THE WRITE CHECK ERROR FLAG
4981 026036 012737 026112 001206 MOV 2#S,SESCAPE ;ESCAPE TO 8S ON ERROR
4982 026044 013737 004176 001270 MOV 2#DTADPB+12,2#CYL.DS ;CYLINDER
4983 026052 113737 004175 001274 MOV 2#DTADPB+11,2#TRK.DS ;TRACK
4984 026060 113737 004174 001272 MOV 2#DTADPB+10,2#SEC.DS ;SECTOR
4985 026066 012746 004202 MOV 2#DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
4986 026072 004737 026216 JSR PC,2#ERINDX ;FORM DISPATCH INDEX
4987 026076 062607 ADD (SP)+,PC ;REPORT PROPER ERROR
4988 026100 104041 ERROR 41 ;
4989 026102 104042 ERROR 42 ;PARITY ERROR
4990 026104 104043 ERROR 43 ;UNSAFE ERROR
4991 026106 104044 ERROR 44 ;NON-I/O ERROR
4992 026110 104046 ERROR 46 ;FATAL WRITE CHECK
4993 026112 013746 004202 8S: MOV DTADPB+16,-(SP) ;STATUS WORD
4994 026116 004737 026156 JSR PC,LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
4995 026122 123737 001364 001103 12S: CMPB 2#ERR.CT,2#SERFLG ;GO TO NEXT TEST?
4996 026130 101002 BHI 10S ;NO--BRANCH
4997 026132 013700 001252 9S: MOV 2#BYPASS,RO ;YES--GET EXIT ADDRESS
4998 026136 032737 040000 001220 10S: BIT 2#SW14,2#C.SWR ;STALL?
4999 026144 001403 BEQ 11S ;NO--BRANCH
5000 026146 004037 026334 JSR RD,2#STALL ;YES--CALL STALL ROUTINE
5001 026152 001356 .WORD STALL2 ;STALL TIME POINTER
5002 026154 000200 11S: RTS RO
5003
5004 ;THIS SUBROUTINE CHECK FOR LOOP, ABORT, OR CONTINUE SWITCHES AFTER
5005 ;ERRORS 41, 42, 43, 44, 45, AND 46.
5006 ;CALL
5007 ; MOV DTA+16,-(SP) ;STATUS WORD FROM DPB IN USE
5008 ; JSR PC,LOP.CK
5009 ; RETURN
5010
5011 026156 032777 001000 152754 LOP.CK: BIT 2#SW9,2#SWR ;LOOP ON ERROR
5012 026164 001402 BEQ 1S ;BR IF NOT
5013 026166 000177 152716 JMP 2#SLPERR ;START AT THE LOOP ADDRESS
5014 026172 005037 001206 1S: CLR SESCAPE ;CLEAR ERROR ESCAPE FLAG
5015 026176 032766 072006 000002 BIT 2#BIT14:BIT13:BIT12:BIT10:BIT02:BIT01,2(SP) ;CHECK ERROR TYPE
5016 026204 001402 BEQ 2S ;BR IF DRIVE NOT OFFLINE, UNLOADED, OR
5017 ;PERSISTENT UNSAFE OR FATAL MASSBUS PARITY
5018 026206 000137 017356 JMP SEOP ;TERMINATE DRIVE
5019 026212 012616 2S: MOV (SP)+,(SP) ;ADJUST RETURN ADDRESS
5020 026214 000207 RTS PC
5021
5022 ;THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
5023 ;TO THE PROPER ERROR CALL. THE INDEX IS FORMED BY EXAMINING
5024 ;THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB.
5025 ;INDEX STATUS/ERROR
5026 ;-----
    
```



```

5027      :      0 BIT14!BIT13!BIT08!BIT01
5028      :      2 BIT11!BIT10!BIT02
5029      :      4 BIT12!BIT04
5030      :      6 BIT05!BIT03!<BIT09 & COMMAND=NON-I/O>
5031      :      10 BIT06!<BIT09 & COMMAND=I/O>
5032      :CALL
5033      :      JSR      @DPB+16,-(SP)      ;ADDRESS OF STATUS/ERROR INDICATOR
5034      :      JSR      PC,@ERINDX      ;FORM INDEX
5035      :      RETURN     ;INDEX IS ON THE STACK
5036
5037      026216 010046      ERINDX: MOV      RO,-(SP)      ;SAVE RO
5038      026220 010146      MOV      R1,-(SP)      ;SAVE R1
5039      026222 016600 000006  MOV      6(SP),RO      ;GET STATUS/ERROR INDICATOR POINTER
5040      026226 011037 001260  MOV      (RO),@SVSTAT ;SAVE THE STATUS/ERROR INDICATOR
5041      026232 005001      CLR      R1            ;START INDEX AT ZERO
5042      026234 032710      BIT      (PC)+(RO)    ;FORM INDEX OF 0?
5043      026236 020402      .WORD   BIT13!BIT08!BIT01
5044      026240 001027      BNE     55            ;YES--BRANCH
5045      026242 032710      BIT      (PC)+(RO)    ;FORM PARITY ERROR OR PORT REQUEST INDEX (2)?
5046      026244 006004      .WORD   BIT11!BIT10!BIT02
5047      026246 001023      BNE     45            ;YES--BRANCH
5048      026250 032710      BIT      (PC)+(RO)    ;FORM UNSAFE INDEX (4)?
5049      026252 050020      .WORD   BIT14!BIT12!BIT04
5050      026254 001017      BNE     35            ;YES--BRANCH
5051      026256 032710      BIT      (PC)+(RO)    ;FORM NON-I/O ERROR INDEX (6)?
5052      026260 000050      .WORD   BIT05!BIT03
5053      026262 001013      BNE     25            ;YES--BRANCH
5054      026264 032710      BIT      (PC)+(RO)    ;FORM I/O ERROR INDEX (10)?
5055      026266 000100      .WORD   BIT06
5056      026270 001007      BNE     15            ;YES--BRANCH
5057      026272 032710      BIT      (PC)+(RO)    ;SOFTWARE TIMEOUT?
5058      026274 001000      .WORD   BIT09
5059      026276 001410      BEQ     55            ;NO--FORM INDEX OF 0
5060      026300 122760 000150 177762  CMPB   #150,-16(RO) ;YES--I/O?
5061      026306 003001      BGT     25            ;NO--BRANCH
5062      026310 005201      15:    INC      R1            ;INDEX=10---ERROR=45 OR 46
5063      026312 005201      25:    INC      R1            ;INDEX=6---ERROR=44
5064      026314 005201      35:    INC      R1            ;INDEX=4---ERROR=43
5065      026316 005201      45:    INC      R1            ;INDEX=2---ERROR=42
5066      026320 006301      55:    ASL     R1            ;INDEX=0---ERROR=41
5067      026322 010166 000006  MOV      R1,6(SP)    ;RETURN INDEX TO USER
5068      026326 012601      MOV      (SP)+,R1    ;RESTORE R1
5069      026330 012600      MOV      (SP)+,RO    ;RESTORE RO
5070      026332 000207      RTS      PC          ;RETURN FROM CALL
5071
5072      :THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
5073      :AMOUNT OF TIME IF BIT13 OF C.SWR = 0 OR A RANDOM AMOUNT OF TIME
5074      :IF BIT 13 OF C.SWR = 1.
5075      :STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0 - 7, AND STALL2
5076      :CONTAINS THE TIME FOR TESTS 16-21.
5077      :CALL
5078      :      JSR      RO,@STALL
5079      :      TIME POINTER      ;WHERE TO FIND THE STALL TIME
5080
5081      026334 013046      STALL: MOV      @<(RO)+,-(SP) ;PICKUP STALL TIME
5082      026336 032737 020000 001220  BIT      #SW13,@C.SWR ;USE A RANDOM TIME?
    
```



```

5083 026344 001406          BEQ      1$
5084 026346 004737 023460    JSR      PC,@$SRAND
5085 026352 013716 023560    MOV      @$LONUM,(SP)
5086 026356 042716 177700    BIC      @C77,(SP)
5087 026362 005046          CLR      -(SP)
5088 026364 162766 000001 000002 1$:  SJB      @1,2(SP)
5089 026372 103407          BLO      4$
5090 026374 012716 000144    MOV      @100.,(SP)
5091 026400 005700          3$:  TST      R0
5092 026402 005366 000000    DEC      0(SP)
5093 026406 001374          BNE      3$
5094 026410 000765          BR       2$
5095 026412 022626          4$:  CMP      (SP)+,(SP)+
5096 026414 000200          RTS      R0

```

```

:NO--BRANCH
:YES--FORM RANDOM NUMBER
:AND USE IT FOR THE STALL TIME
:BUT NEVER > 64 MILLISECONDS
:CLEAR TEMP. LOCATION
:MORE STALL REQUIRED?
:NO--BRANCH
:STALL FOR ABOUT 1 MILLISECOND
:NOP TO KILL TIME
:COUNT
:LOOP IF MORE COUNTS NEEDED
:CLEAN OFF THE STACK
:EXIT

```

```

5099          :ROUTINE TO PROVIDE A 2 MS STALL AFTER A SEEK OPERATION IN THE SEEK TIMING
5100          :TESTS. THIS STALL IS REQUIRED TO COMPENSATE FOR THE 'ACCESS READY' DELAY
5101          :IN THE RPO4. THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES.
5102          :CALL

```

```

5103          :      JSR      PC,@$TWOMS
5104          :      RETURN

```

```

5106 026416 013746 177776          TWOMS:  MOV      @$PS,-(SP)
5107 026422 012737 000240 177776    MOV      @(<5*32.)@$PS
5108 026430 017746 152740    MOV      @$PKV,-(SP)
5109 026434 012777 026460 152732    MOV      @1$,$PKV
5110 026442 012777 000310 152732    MOV      @200.,$PKB
5111 026450 012777 000101 152722    MOV      @101,$PKCS
5112 026456 000001          WAIT
5113 026460 062706 000004          1$:  ADD      @4,SP
5114 026464 012677 152704    MOV      (SP)+,$PKV
5115 026470 012637 177776    MOV      (SP)+,$PS
5116 026474 000207          RTS      PC

```

```

:SAVE THE PRESENT PROCESSOR STATUS
:SET THE PROCESSOR PRIORITY TO 5
:SAVE THE OLD CLOCK VECTOR ADDRESS
:SETUP NEW VECTOR ADDRESS
:LOAD THE CLOCK BUFFER
:START THE CLOCK
:WAIT FOR 2 MS
:INCREMENT STACK FOR RETURN
:RESTORE OLD CLOCK VECTOR
:RESTORE THE OLD PROCESSOR STATUS
:RETURN

```

```

5118          :ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEEKS
5119          :CALL

```

```

5120          :      JSR      R0,$VERIFY
5121          :      ADR POINTER
5122          :      RETURN

```

```

5124 026476 010146          VERIFY:  MOV      R1,-(SP)
5125 026500 012001          MOV      (R0)+,R1
5126 026502 042737 010000 047662    BIC      @$FMT22,$BUFFER
5127 026510 023761 047662 000002    CMP      @$BUFFER,2(R1)
5128 026516 001003          BNE      1$
5129 026520 023711 047664          CMP      @$BUFFER+2,(R1)
5130 026524 001441          BEQ      3$
5131 026526 013737 047662 001262 1$:  MOV      @$BUFFER,$CYL,R0
5132 026534 113737 047665 001264    MOV      @$BUFFER+3,$TRK,R0
5133 026542 113737 047664 001266    MOV      @$BUFFER+2,$SEC,R0
5134 026550 112137 001272    MOV      (R1)+,$SEC,DS
5135 026554 112137 001274    MOV      (R1)+,$TRK,DS
5136 026560 011137 001270    MOV      (R1), $CYL,DS
5137 026564 012737 026576 001206    MOV      @2$, $ESCAPE
5138 026572 005740          TST      -(R0)

```

```

:ADDRESS OF DPB+10 (SECTOR NUMBER)
:SAVE R1
:GET ADDRESS OF DPB+10
:STRIP FORMAT BIT FROM CYLINDER NUMBER
:CYLINDER NUMBER OK?
:NO--BRANCH
:YES--HOW ABOUT TRACK/SECTOR?
:BRANCH IF GOOD
:SAVE THE EXPECTED AND THE
:RECEIVED CYLINDER, TRACK,
:AND SECTOR
:; ESCAPE TO 2$ ON ERROR
:MAKE IT TEST PC+4

```



```

S139 026574 104012          ERROR 12          ;REPORT THE ERROR
S140 026576 012737 000107 004106 25:  MOV  #RECAL,DPB,A+2 ;LOAD RECALIBRATE ORDER CODE
S141 026604 004037 025002          JSR  RD,#CALL.A   ;GO EXECUTE THE COMMAND
S142 026610 005037 001206          CLR  $ESCAPE     ;CLEAR ERROR ESCAPE FLAG
S143 026614 032777 001000 152316  BIT  #SW9,#SWR   ;LOOP ON ERROR ?
S144 026622 001404          BEQ  45          ;BR IF NOT
S145 026624 000177 152260          JMP  #SLPERR    ;RETURN TO ERROR LOOP ADDRESS
S146 026630 062700 000002          35:  ADD  #2,RD    ;INCREMENT RETURN ADDRESS
S147 026634 012601          45:  MOV  (SP)+,R1  ;RESTORE R1
S148 026636 000200          RTS  RD         ;EXIT

; THIS ROUTINE WILL PERFORM A "MASSBUS" INIT. FOLLOWED BY
; A "RECALIBRATE" ON THE DRIVE UNDER TEST.
; NOTE: THIS ROUTINE DESTROYS R1 AND R4
; CALL
; JSR  RD,SRCH00 ;DO A MASSBUS INIT. AND RECAL
; RETURN1 ;RETURN HERE IF NO ERROR
; RETURN2 ;RETURN HERE ON ERROR
;
SRCH00: CLR  R1 ;INCASE OF ERROR (TYPTIM)
        CLR  #PS
        MOV  #ISR,#RVEC ;SETUP INTERRUPT VECTOR
        MOV  #RPA0,R4 ;PICKUP ADDRESS OF RPCS1
        MOV  #BIT05,RPCS2(R4) ;MASSBUS INIT.
        CLR  #DTADPB+10 ;TRACK=0; SECTOR=0
        CLR  #DTADPB+12 ;CYLINDER =0
        MOV  #RECAL,#DTADPB+2 ;COMMAND = RECALIBRATE
        CLR  #SEESCAPE ;NO ESCAPE ADDRESS
        JSR  RD,#RPO4 ;CALL THE DRIVER
        DTADPB ;DPB POINTER
        BR  45 ;QUEUE IS FULL
        15:  TST  DTADPB+16 ;WAIT ON DONE
        BEQ  15
        BPL  35 ;TAKE NORMAL EXIT IF NO ERROR
        MOV  #25,$ESCAPE ;ESCAPE TO 25 ON ERROR
        MOV  #DTADPB+12,#CYL.DS ;CYLINDER
        MOVB #DTADPB+11,#TRK.DS ;TRACK
        MOVB #DTADPB+10,#SEC.DS ;SECTOR
        MOV  #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
        JSR  PC,#ERINDX ;FORM DISPATCH INDEX
        ADD  (SP)+,PC ;REPORT PROPER ERROR
        ERROR 41
        ERROR 42 ;PARITY ERROR
        ERROR 43 ;UNSAFE ERROR
        ERROR 44 ;NON-I/O ERROR
        ERROR 45 ;I/O ERROR
        25:  TST  (RD)+ ;ADJUST FOR ERROR EXIT
        BR  45 ;GO TO THE EXIT
        35:  CLR  RPOA(R4) ;TRACK AND SECTOR = 0
        CLR  RPCA(R4) ;CYLINDER = 0
        45:  RTS  RD ;RETURN

; THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS & THE SERVO SETTLE DOWN TEST
DORTI: RTI ;RETURN FROM INTERRUPT
S193 027022 000002
S194

```


5195
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250

027024 104412
027026 012700 001276
027032 012701 001332
027036 005020
027040 020001
027042 103775
027044 012710 047662
027050 012737 077777 001276
027056 012737 077777 001314
027064 104413
027066 000207

012702 001276
005705
001402
012702 001314
027722 152274
002003
017762 152266 177776
027763 152260 000004
002001
005212
005722
027722 152244
003403
017762 152236 177776
027763 152230 000006
003401
005212
005722
067722 152214
005522
005212
022737 056152 001332
101406
017777 152174 152120
062737 000002 001332
000207

: THIS ROUTINE WILL INITIALIZE THE TIMERS USED BY THE "TIMING ROUTINES"

```
CALL JSR PC, @STRTMR
:
STRTMR: SAVREG ; SAVE RD-RS
MOV @TIM.UP, R0 ; START AT TIM.UP (MINIMUM)
MOV @TIM.PT, R1 ; STOP AT TIM.PT
1S: CLR (R0)+ ; CLEAR
CMP R0, R1 ; DONE?
BLO 1S ; NO--BRANCH
MOV @BUFFER, (R0) ; SETUP POINTER
MOV @CBIT15, @TIM.UP ; SET MINIMUM TIME TO MAXIMUM
MOV @CBIT15, @TIM.DN ; POSITIVE NUMBER
RESREG ; RESTORE RD-RS
RTS PC ; RETURN
```

: THIS ROUTINE WILL ADD THE ELAPSED TIME TO THE AVERAGE COUNTER AND MAINTAIN THE MINIMUM AND MAXIMUM TIMES.

```
NOTE: THIS ROUTINE DESTROYS R2
CALL MOV @TP, R3 ; PARAMETER POINTER
MOV FLAG, R5 ; FLAG=0=COUNT UP
; FLAG=-1=COUNT DOWN
:
JSR PC, @COUNT
RETURN
COUNT: MOV @TIM.UP, R2 ; PICKUP THE "UP" POINTER
TST R5 ; USE IT?
BEQ 1S ; YES--BRANCH
MOV @TIM.DN, R2 ; NO--PICKUP "DOWN" POINTER
1S: CMP @PKC, (R2)+ ; LESS THAN PREVIOUS LOW?
BGE 2S ; NO--BRANCH
MOV @PKC, -2(R2) ; YES--SAVE IT
2S: CMP @PKC, 4(R3) ; LESS THAN THE LOW LIMIT?
BGE 3S ; NO--BRANCH
INC (R2) ; YES--COUNT IT
3S: TST (R2)+ ; ADVANCE THE POINTER
CMP @PKC, (R2)+ ; GREATER THAN PREVIOUS HIGH?
BLE 4S ; NO--BRANCH
MOV @PKC, -2(R2) ; YES--SAVE IT
4S: CMP @PKC, 6(R3) ; GREATER THAN THE HIGH LIMIT?
BLE 5S ; NO--BRANCH
INC (R2) ; YES--COUNT IT
5S: TST (R2)+ ; ADVANCE THE POINTER
ADD @PKC, (R2)+ ; ADD THIS COUNT TO THE TOTAL
ADC (R2)+
INC (R2)
6S: CMP @BUFFER+(<4*814.>), @TIM.PT ; COUNT THIS READING
BLOS 6S ; NO--BRANCH
MOV @PKC, @TIM.PT ; YES--WELL SAVE IT THEN
ADD @2, @TIM.PT ; ADVANCE THE POINTER
RTS PC ; RETURN
```

: THIS ROUTINE IS USED TO TYPE THE MINIMUM, MAXIMUM, AND AVERAGE TIMES FOR THE TIMING TESTS

5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306

```

027222 012002
027224 032777 000100 151706
027232 001145
027234 012237 027254
027240 012205
027242 012203
027244 011202
027246 012704 001276
027252 104401
027254 000000
027256 005764 000014
027262 001527
027264 104401 043650
027270 012446
027272 004737 023170
027276 004737 023420
027302 104401 043675
027306 005724
027310 001421
027312 104401 044010
027316 016446 177776
027322 004737 023170
027326 004737 023420
027332 104401 043702
027336 010346
027340 004737 023170
027344 004737 023420
027350 104401 043675
027354 104401 043657
027360 012446
027362 004737 023170
027366 004737 023420
027372 104401 043675
027376 005724
027400 000421
027402 104401 044010
027406 016446 177776
027412 004737 023170
027416 004737 023420
027422 104401 043731
027426 010246
027430 004737 023170
027434 004737 023420
    
```

```

: IT WILL ALSO CHECK THE TIMES TO ENSURE
: THEY ARE WITHIN TOLERANCE AND IF NOT FLAG THE BAD TIMES.
: NOTE: THIS ROUTINE DESTROYS R2-R5
CALL
      JSR      RD,2#TYPTIM      ;GO REPORT THE TIMES
      TABLE  ;POINT TO THE PROPER TABLE
      RETURN
:
: TABLE:MSGADR1      ;ADDRESS OF ASCIZ MESSAGE NUMBER 1
: MSGADR2           ;ADDRESS OF ASCIZ MESSAGE NUMBER 2
: MIN.ALLOWED       ;MINIMUM TIME ALLOWED
: MAX.ALLOWED       ;MAXIMUM TIME ALLOWED
:
TYPTIM: MOV      (R0)+,R2      ;PICKUP THE TABLE POINTER
        BIT      #SMD6,2SMR   ;INHIBIT TIME REPORTS?
        BNE     7$          ;YES--BRANCH
        MOV     (R2)+,2$     ;ADDRESS OF MESSAGE NUMBER 1
        MOV     (R2)+,R5     ;ADDRESS OF MESSAGE NUMBER 2
        MOV     (R2)+,R3     ;PICKUP THE LOW LIMIT
        MOV     (R2),R2      ;AND THE HIGH LIMIT
        MOV     #TIM,UP,R4   ;PARAMETER POINTER
1$:     TYPE     ;TYPE THE MESSAGE
2$:     .WORD   0            ;ASCIZ MESSAGE POINTER GOES HERE
        TST     14(R4)      ;DID ANY COUNTS OCCUR?
        BEQ     6$          ;NO--BRANCH
        TYPE   ,MSGMIN     ;"MIN="
        MOV     (R4)+,-(SP) ;PUT (R4)+ ON THE STACK
        JSR    PC,2#SSB2D  ;CHANGE (R4)+ TO DECIMAL ASCIZ
        JSR    PC,2#SSUPRS ;TYPE WITHOUT LEADING ZEROS
        TYPE   ,MSGOUS     ;"0 US"
        TST     (R4)+      ;ANY SEEKS BELOW THE LOW LIMIT
        BEQ     3$          ;NO--BRANCH
        TYPE   ,MSG.SP     ;
        MOV     -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
        JSR    PC,2#SSB2D  ;CHANGE -2(R4) TO DECIMAL ASCIZ
        JSR    PC,2#SSUPRS ;TYPE WITHOUT LEADING ZEROS
        TYPE   ,MBELOW    ;"BELOW THE MINIMUM OF"
        MOV     R3,-(SP)   ;PUT R3 ON THE STACK
        JSR    PC,2#SSB2D  ;CHANGE R3 TO DECIMAL ASCIZ
        JSR    PC,2#SSUPRS ;TYPE WITHOUT LEADING ZEROS
        TYPE   ,MSGOUS     ;
        TST     (R4)+      ;ANY SEEKS ABOVE THE HIGH LIMIT
        BR     4$          ;NO--BRANCH
        TYPE   ,MSG.SP     ;YES--REPORT HOW MANY
        MOV     -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
        JSR    PC,2#SSB2D  ;CHANGE -2(R4) TO DECIMAL ASCIZ
        JSR    PC,2#SSUPRS ;TYPE WITHOUT LEADING ZEROS
        TYPE   ,MABOVE    ;"ABOVE THE MAXIMUM OF"
        MOV     R2,-(SP)   ;PUT R2 ON THE STACK
        JSR    PC,2#SSB2D  ;CHANGE R2 TO DECIMAL ASCIZ
        JSR    PC,2#SSUPRS ;TYPE WITHOUT LEADING ZEROS
    
```



```

5307 027440 104401 043675
5308 027444 104401 043666
5309 027450 012446
5310 027452 012446
5311 027454 012446
5312 027456 004737 023562
5313 027462 006126
5314 027464 100001
5315 027466 005216
5316 027470
5317 027470 004737 023170
5318 027474 004737 023420
5319 027500 104401 043675
5320 027504 104401 044010
5321 027510 016446 177776
5322 027514 004737 023170
5323 027520 004737 023420
5324 027524 104401 043760
5325 027530 010537 027254
5326 027534 001404
5327 027536 005005
5328 027540 000644
5329 027542 104401 043775
5330 027546 000200
5331
5332
5333
5334
5335
5336
5337
5338
5339 027550 020237 001520
5340 027554 001410
5341 027556 063702 001522
5342 027562 020237 001520
5343 027566 003402
5344 027570 013702 001520
5345 027574 005720
5346 027576 000200
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356 027600 020137 001512
5357 027604 001410
5358 027606 063701 001514
5359 027612 020137 001512
5360 027616 003402
5361 027620 013701 001512
5362 027624 005720

4S:  TYPE      ,MSGOUS
      TYPE      ,MSGAVG      ;"AVG="
      MOV      (R4)+,-(SP)    ;FORM THE AVERAGE
      MOV      (R4)+,-(SP)
      MOV      (R4)+,-(SP)
      JSR      PC,2#SDIV
      ROL      (SP)+
      BPL      5S
      INC      (SP)
      ;IS THE REMAINDER OVER HALF?
      ;NO--BRANCH
      ;YES--ROUND UP

5S:  JSR      PC,2#SSB2D      ;CHANGE TO DECIMAL ASCIZ
      JSR      PC,2#SSUPRS    ;TYPE WITHOUT LEADING ZEROS
      TYPE      ,MSGOUS
      TYPE      ,MSG.SP
      MOV      -2(R4),-(SP)   ;PUT -2(R4) ON THE STACK
      JSR      PC,2#SSB2D    ;CHANGE -2(R4) TO DECIMAL ASCIZ
      JSR      PC,2#SSUPRS    ;TYPE WITHOUT LEADING ZEROS
      TYPE      ,MSGNUM
      MOV      R5,2S
      BEQ      7S
      CLR      R5
      BR       1S
      ;"SEEKS TIMED"
      ;NEXT MESSAGE POINTER
      ;IF NONE EXIT
      ;NO MORE THAN 2

6S:  TYPE      ,MSGNON
7S:  RTS      R0
      ;EXIT

;THIS SUBROUTINE WILL INCREMENT THE TRACK
;NUMBER (R2) BY THE AMOUNT SPECIFIED BY 'IT'.
;CALL
;
;      JSR      R0,2#INCTRK
;      RETURN1
;      RETURN2
;
;      ;TRACK NUMBER GREATER THAN LT15
;      ;TRACK NUMBER INCREMENTED

INCTRK: CMP      R2,2#LT
        BEQ      2S
        ADD      2#IT,R2
        CMP      R2,2#LT
        BLE      1S
        MOV      2#LT,R2
        ;LAST TRACK COMPLETED?
        ;YES--EXIT
        ;NO--UPDATE TRACK
        ;TRACK TO BIG?
        ;NO--EXIT
        ;YES--SET TRACK TO LAST TRACK
1S:  TST      (R0)+
        ;ADJUST FOR RETURN 2
2S:  RTS      R0
        ;RETURN

;THIS SUBROUTINE WILL INCREMENT THE CYLINDER
;NUMBER (R1) BY THE AMOUNT SPECIFIED BY 'IC'.
;CALL
;
;      JSR      R0,2#INCCYL
;      RETURN1
;      RETURN2
;
;      ;CYLINDER NUMBER GREATER THAN LC15
;      ;CYLINDER NUMBER INCREMENTED

INCCYL: CMP      R1,2#LC
        BEQ      2S
        ADD      2#IC,R1
        CMP      R1,2#LC
        BLE      1S
        MOV      2#LC,R1
        ;LAST CYLINDER COMPLETED?
        ;YES--EXIT
        ;NO--UPDATE CYLINDER
        ;CYLINDER TO BIG?
        ;NO--EXIT
        ;YES--SET CYLINDER TO LAST CYLINDER
1S:  TST      (R0)+
        ;ADJUST FOR RETURN 2
    
```



```

5363 027626 000200 2S: RTS RO ;RETURN
5364
5365 :THIS ROUTINE DECREASES THE SECTOR ADDRESS.
5366 :CALL
5367 : CLR -(SP) ;CLEAR THE STACK
5368 : JSR PC,DECSEC ;SUBROUTINE ENTRY
5369 : RETURN
5370
5371 027630 113766 004212 000002 DECSEC: MOVB RP,REG+RPDA,2(SP) ;PUT THE SECTOR ADDRESS ON THE STACK
5372 027636 005366 000002 DEC 2(SP) ;DECREMENT THE ADDRESS
5373 027642 100003 BPL 1S ;BR IF NOT CORRECTION NEEDED
5374 027644 013766 001634 000002 1S: MOV PRMLMT+22.,2(SP) ;OVERFLOW OCCURED, FORCE TO MAXIMUM ADDRESS
5375 027652 000207 2S: RTS PC ;RETURN
5376
5377 :THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
5378 :WITH ADDRESSES FROM 0 TO 21 WITH EACH ADDRESS
5379 :BEING STORED IN 256 CONSECUTIVE LOCATIONS
5380 :CALL
5381 : JSR PC,#FILBUF
5382 : RETURN
5383
5384 027654 104412 FILBUF: SAVREG ;SAVE RO - RS
5385 027656 005000 CLR RO ;FIRST DISK ADDRESS
5386 027660 012701 047662 MOV #BUFFER,R1 ;START FILLING HERE
5387 027664 012702 000400 1S: MOV #256,R2 ;DO 256 WORDS
5388 027670 010021 2S: MOV RO,(R1)+ ;STORE
5389 027672 005302 DEC R2 ;MORE?
5390 027674 003375 BGT 2S ;YES--BRANCH
5391 027676 005200 INC RO ;NO--UPDATE DISK ADDRESS
5392 027700 023700 001630 CMP PRMLMT+22,RO ;DONE?
5393 027704 103367 BHIS 1S ;NO--BRANCH
5394 027706 104413 RESREG ;RESTORE RO - RS
5395 027710 000207 RTS PC ;RETURN
5396
5397 :THIS ROUTINE WILL CLEAR THE BUFFER BY
5398 :SETTING EACH WORD TO "177400".
5399 :CALL
5400 : JSR RO,#CLRBUF
5401 : RETURN
5402
5403 027712 104412 CLRBUF: SAVREG ;SAVE RO - RS
5404 027714 012701 177400 MOV #177400,R1 ;WORD TO FILL BUFFER WITH
5405 027720 012702 047662 MOV #BUFFER,R2 ;FIRST ADDRESS OF BUFFER
5406 027724 012703 075662 MOV #BUFFER+(512.*22.),R3 ;LAST ADDRESS+2 OF BUFFER
5407 027730 010122 1S: MOV R1,(R2)+ ;FILL WORDS 1, 9,...249,...5625
5408 027732 010122 MOV R1,(R2)+ ;FILL WORDS 2,10,...250,...5626
5409 027734 010122 MOV R1,(R2)+ ;FILL WORDS 3,11,...251,...5627
5410 027736 010122 MOV R1,(R2)+ ;FILL WORDS 4,12,...252,...5628
5411 027740 010122 MOV R1,(R2)+ ;FILL WORDS 5,13,...253,...5629
5412 027742 010122 MOV R1,(R2)+ ;FILL WORDS 6,14,...254,...5630
5413 027744 010122 MOV R1,(R2)+ ;FILL WORDS 7,15,...255,...5631
5414 027746 010122 MOV R1,(R2)+ ;FILL WORDS 8,16,...256,...5632
5415 027750 020203 CMP R2,R3 ;DONE?
5416 027752 103766 BLO 1S ;NO--BRANCH
5417 027754 104413 RESREG ;RESTORE RO - RS
5418 027756 000200 RTS RO ;RETURN FROM CALL
    
```



```

5419
5420
5421
5422
5423
5424
5425
5426
5427 027760 104412
5428 027762 162706 000004
5429 027766 005001
5430 027770 012716 047662
5431 027774 005066 000002
5432 030000 012702 000020
5433 030004 011603
5434 030006
5435 030006 020123
5436 030010 001063
5437 030012 020123
5438 030014 001061
5439 030016 020123
5440 030020 001057
5441 030022 020123
5442 030024 001055
5443 030026 020123
5444 030030 001053
5445 030032 020123
5446 030034 001051
5447 030036 020123
5448 030040 001047
5449 030042 020123
5450 030044 001045
5451 030046 020123
5452 030050 001043
5453 030052 020123
5454 030054 001041
5455 030056 020123
5456 030060 001037
5457 030062 020123
5458 030064 001035
5459 030066 020123
5460 030070 001033
5461 030072 020123
5462 030074 001031
5463 030076 020123
5464 030100 001027
5465 030102 020123
5466 030104 001025
5467 030106 005302
5468 030110 001336
5469 030112 062716 001000
5470 030116 005201
5471 030120 023701 001630
5472 030124 103325
5473 030126 005766 000002
5474 030132 001406
    
```

```

; THIS ROUTINE IS USED TO CHECK THE DATA BUFFER
; FOR ADDRESSES 0 THROUGH 21 WITH EACH ADDRESS
; BEING STORED IN 256 CONSECUTIVE LOCATIONS
CALL
; JSR      RO, @CKSCTR
; RETURN
CKSCTR: SAVREG
; SAVE R0 - R5
SUB      #4, SP ; RESERVE TEMP. STORAGE AREA
CLR      R1     ; FIRST SECTOR
MOV      #BUFFER, (SP) ; FIRST ADDRESS OF DATA BUFFER
CLR      2(SP)  ; NO ERRORS
15:      MOV      #16, R2 ; LOOP COUNT (16*16=256)
        MOV      (SP), R3 ; GET 1ST ADDRESS OF THIS SECTORS DATA
25:      CMP      R1, (R3)+ ; WORD 1
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 2
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 3
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 4
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 5
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 6
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 7
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 8
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 9
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 10
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 11
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 12
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 13
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 14
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 15
        BNE      7$      ; BRANCH IF BAD
        CMP      R1, (R3)+ ; WORD 16
        BNE      7$      ; BRANCH IF BAD
        DEC      R2      ; FINISHED WITH THIS SECTORS DATA?
        BNE      25$    ; NO--BRANCH
35:      ADD      #512., (SP) ; YES--FIRST ADDRESS OF NEXT SECTOR
        INC      R1     ; MOVE TO NEXT SECTOR
        CMP      PRMLMT+22, R1 ; DONE?
        BHS     1$      ; NO--BRANCH
45:      TST      2(SP)  ; ERROR OCCUR?
        BEQ     6$      ; NO--BRANCH
    
```


BUJ

```

5475 030134 123737 001364 001103      CMPB    2#ERR.CT,2#SERFLG ;MAX. ERROR OCCURRED?
5476 030142 101002                      BHI     6$
5477 030144 013700 001252      5$:    MOV    2#BYPASS,R0 ;TAKE ERROR EXIT
5478 030150 062706 000004      6$:    ADD    #4,SP ;FREE TEMP. AREA
5479 030154 104413                      RESREG ;RESTORE R0 - R5
5480 030156 000200                      RTS     R0 ;RETURN FROM CALL
5481 030160 010304      7$:    MOV    R3,R4 ;FORM WORD NUMBER AND
5482 030162 161604                      SUB    (SP),R4 ;ADDRESS TO CONTINUE FROM
5483 030164 010415                      MOV    R4,R5
5484 030166 006204                      ASR    R4 ;WORD NUMBER
5485 030170 042705 177740                      BIC    #1C37,R5
5486 030174 001002                      BNE    8$ ;BRANCH IF NOT A MULTIPLE OF 16
5487 030176 012705 000040      8$:    MOV    #40,R5 ;SET TO WORD 16
5488 030202 006305                      ASL    R5
5489 030204 062705 030006      9$:    ADD    #2$ ,R5 ;ADDRESS
5490 030210 016337 177776 001126      MOV    -2(R3),2#BDDAT ;SAVE BAD DATA
5491 030216 005766 000002      TST    2(SP) ;FIRST ERROR?
5492 030222 001015                      BNE    10$ ;NO--BRANCH
5493 030224 013737 004176 001270      MOV    2#DTADPB+12,2#CYL.DS ;CYLINDER NUMBER
5494 030232 113737 004175 001274      MOV    2#DTADPB+11,2#TRK.DS ;TRACK NUMBER
5495 030240 012737 030250 001206      MOV    #9$,SESCAPE ;ESCAPE TO 9$ ON ERROR
5496 030246 104021                      ERROR  21 ;REPORT THE ERROR
5497 030250 105166 000002      9$:    COMB  2(SP) ;SET ERROR SWITCH
5498 030254 000404                      BR     11$
5499 030256                      10$:
5500 030256 012737 030266 001206      MOV    #11$,SESCAPE ;ESCAPE TO 11$ ON ERROR
5501 030264 104022                      ERROR  22 ;REPORT THE ERROR
5502 030266 032777 001000 150644      11$:  BIT    #SM09,2$MR ;LOOP ON ERROR?
5503 030274 001323                      BNE    5$ ;YES
5504 030276 032777 000002 150634      BIT    #SM01,2$MR ;STOP DATA COMPARE?
5505 030304 001310                      BNE    4$ ;YES--BRANCH
5506 030306 123737 001364 001103      CMPB    2#ERR.CT,2#SERFLG ;MAX. ERRORS?
5507 030314 101713                      BLOS   5$ ;YES--BRANCH
5508 030316 032777 000040 150614      BIT    #SM05,2$MR ;REPORT ONLY 1ST ERROR PER SECTOR?
5509 030324 001272                      BNE    3$ ;YES--BRANCH
5510 030326 000115                      JMP    (R5)

; THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
; DESIRED PATTERN INTO THE DATA BUFFER.
; CALL
;     MOV    #N$,R0 ;PATTERN NUMBER INDEX TO R0
;     JSR    PC,2#SETBUF
;
; SETBUF: SAVREG ;SAVE R0 - R5
;     MOV    #BUFFER,R1 ;FIRST ADDRESS
;     MOV    2#DTADPB+4,R2 ;WORD COUNT
;     MOV    PAT.PT(R0),R3 ;PICKUP PATTERN POINTER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 1 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 2 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 3 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 4 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 5 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 6 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 7 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 8 INTO DATA BUFFER
;     MOV    (R3)+,(R1)+ ;MOVE WORD 9 INTO DATA BUFFER
5511
5512
5513
5514
5515
5516
5517
5518 030330 104412
5519 030332 012701 047662
5520 030336 013702 004170
5521 030342 016003 003044
5522 030346 012321
5523 030350 012321
5524 030352 012321
5525 030354 012321
5526 030356 012321
5527 030360 012321
5528 030362 012321
5529 030364 012321
5530 030366 012321
    
```



```

5587 030536 001026          BNE      4S          ; BRANCH IF DIFFERENT
5588 030540 062702 000020  ADD      #16.,R2     ; DONE ?
5589 030544 001333          BNE      1S          ; NO--BRANCH
5590 030546 005726          TST      (SP)+      ; YES -- CLEAN UP STACK
5591 030550 104413          RESREG          ; RESTORE R0 - R5
5592 030552 000207          RTS             PC
5593 030554 010104          MOV      R1,R4     ; FORM THE WORD NUMBER
5594 030556 162704 047662  SUB      #BUFFER,R4
5595 030562 006204          ASR      R4        ; WORD NUMBER
5596 030564 010305          MOV      R3,R5     ; FORM ADDRESS TO CONTINUE FROM
5597 030566 166005 003044  SUB      PAT.PT(R0),R5
5598 030572 006305          ASL      R5
5599 030574 062705 030440  ADD      #2S,R5     ; ADDRESS
5600 030600 064341          ADD      -(R3),-(R1) ; RECONSTRUCT THE BAD WORD
5601 030602 010137 001122  MOV      R1,#$BDAOR ; SAVE THE ERROR INFORMATION
5602 030606 010337 001120  MOV      R3,#$GDAOR
5603 030612 012137 001126  MOV      (R1)+,#$BDDAT
5604 030616 012337 001124  MOV      (R3)+,#$GDDAT
5605 030622 005716          TST      (SP)
5606 030624 001023          BNE      6S          ; 1ST DATA COMPARE ERROR?
5607 030626 013737 004176 001270  MOV      #DTADPB+12,#CYL.DS ; CYLINDER
5608 030634 113737 004175 001274  MOV      #DTADPB+11,#TRK.DS ; TRACK
5609 030642 113737 004174 001272  MOV      #DTADPB+10,#SEC.DS ; SECTOR
5610 030650 016600 000016  MOV      16(SP),R0 ; GET TEST PC+4
5611 030654 012737 030664 001206  MOV      #5S,SESCAPE ; ESCAPE TO 5S ON ERROR
5612 030662 104013          ERROR      13      ; REPORT THE ERROR
5613 030664 016600 000014          MOV      14(SP),R0 ; PATTERN NUMBER INDEX
5614 030670 105116          COMB      (SP)    ; SET THE ERROR SWITCH
5615 030672 000404          BR       7S
5616 030674          BR       6S
5617 030674 012737 030704 001206  MOV      #7S,SESCAPE ; ESCAPE TO 7S ON ERROR
5618 030702 104014          ERROR      14      ; REPORT THE ERROR
5619 030704 032777 000002 150226 7S:  BIT      #5:01,#SWR ; STOP DATA COMPARE?
5620 030712 001315          BNE      3S          ; YES--EXIT
5621 030714 123737 001364 001103  CMP      #ERR.CT,#SERFLG ; MAX. ERRORS?
5622 030722 101004          BHI      8S          ; NO--BRANCH
5623 030724 013766 001252 000016  MOV      #BYPASS,16(SP) ; YES--ERROR EXIT
5624 030732 000705          BR       3S
5625 030734 000115          BR       8S:      JNP      (R5) ; NO--CONTINUE AT NEXT WORD

; THIS ROUTINE WILL FILL THE DATA BUFFER (256*22 WORDS) WITH
; A RANDOM PATTERN. THE FIRST TWO WORDS OF EVERY 256 WILL
; BE THE BASE OF THE RANDOM NUMBER GENERATOR FOR THE
; NEXT 254 WORDS.
; NOTE: THIS ROUTINE DESTROYS R1 AND R2
; CALL
; JSR      RO,#FILRAN
; RETURN
FILRAN: MOV      #BUFFER,R1
; MAXIMUM NUMBER OF SECTORS
1S:  MOV      PRMLMT+22,R2
; JSR      RO,#RANPAT
DEC      R2
BPL      1S
RTS      R0

```


5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698

030760 013746 023556
 030764 013746 023560
 030770 012702 050662
 030774 012701 051662
 031000 010103
 031002 011237 023560
 031006 016237 000002 023556
 031014 004037 031156
 031020 012637 023560
 031024 012637 023556
 031030 005046
 031032 162322
 031034 001441
 031036 012737 031110 001206
 031044 064342
 031046 010237 001122
 031052 010337 001120
 031056 012237 001126
 031062 012337 001124
 031066 010204
 031070 162704 050662
 031074 006204
 031076 005716
 031100 001002
 031102 105116
 031104 104015
 031106 104016
 031110 032777 001000 150022
 031116 001012
 031120 123737 001364 001103
 031126 101406
 031130 032777 000002 150002
 031136 001002
 031140 020103
 031142 101333
 031144 005726
 031146 001402
 031150 013700 001252
 031154 000200

```

; THIS ROUTINE USES THE FIRST TWO WORDS OF THE
; READ BUFFER TO GENERATE A RANDOM PATTERN. THEN
; THE READ BUFFER IS COMPARED TO THE PATTERN GENERATED.
; NOTE: THIS ROUTINE DESTROYS R1-R4
CALL
      JSR    RO, @RANCK
      RETURN

RANCK: MOV    @SHINUM, -(SP) ; SAVE THE PRESENT RANDOM NUMBER
      MOV    @SLONUM, -(SP)
      MOV    @BUFFER+512., R2 ; READ BUFFER ADDRESS
      MOV    @BUFFER+1024., R1 ; RANDOM PATTERN ADDRESS
      MOV    R1, R3 ; COPY IT INTO R3 FOR LATER USE
      MOV    (R2), @SLONUM ; PRIME THE RANDOM NUMBER GENERATOR
      MOV    2(R2), @SHINUM
      JSR    RO, @RANPAT ; GENERATE A RANDOM PATTERN
      MOV    (SP)+, @SLONUM ; RESTORE PRESENT RANDOM NUMBER
      MOV    (SP)+, @SHINUM
      CLR    -(SP) ; NO ERRORS
      SUB    (R3)+, (R2)+ ; ARE THESE TWO WORDS DIFFERENT?
      BEQ    4$ ; NO--BRANCH
      MOV    @3$, @ESCAPE ; ESCAPE TO 3$ ON ERROR
      ADD    -(R3), -(R2) ; RECREATE THE BAD WORD
      MOV    R2, @BDAOR ; ADDRESS OF BAD DATA
      MOV    R3, @GDAOR ; ADDRESS OF GOOD DATA
      MOV    (R2)+, @BDOAT ; BAD DATA
      MOV    (R3)+, @GDOAT ; GOOD DATA
      MOV    R2, R4 ; FORM WORD NUMBER (1 TO 256)
      SUB    @BUFFER+512., R4
      ASR    R4
      TST    (SP) ; FIRST ERROR
      BNE    2$ ; NO--BRANCH
      COMB   (SP) ; YES--SET ERROR SWITCH
      ERROR 15 ; REPORT THE ERROR
      ERROR 16 ; REPORT THE ERROR
      BIT    @SW09, @SWR ; LOOP ON ERROR?
      BNE    5$ ; YES--BRANCH
      CMPB  @ERR.CT, @SERFLG ; MAX. ERRORS OCCURRED?
      BLOS  5$ ; YES--BRANCH
      BIT    @SW01, @SWR ; STOP COMPARING?
      BNE    5$ ; YES--BRANCH
      CMP    R1, R3 ; ALL DATA BEEN COMPARED?
      BHI    1$ ; NO--BRANCH
      TST   (SP)+ ; ERROR OCCUR?
      BEQ    6$ ; NO--BRANCH
      MOV    @BYPASS, RO ; TAKE ERROR EXIT
      RTS   RO ; EXIT

1$:
2$:
3$:
4$:
5$:
6$:

```

```

; THIS ROUTINE FILLS A 256 WORD BUFFER WITH A RANDOM
; PATTERN OF WHICH THE FIRST TWO WORDS ARE THE BASE
; OF THE PATTERN.
CALL
      MOV    @AOR, R1 ; ADDRESS OF THE BUFFER
      JSR    RO, @RANPAT
      RETURN

```



```

5699 031156 010246
5700 031160 012702 000200
5701 031164 000402
5702 031166 004737 023460
5703 031172 013721 023560
5704 031176 013721 023556
5705 031202 005302
5706 031204 003370
5707 031206 012602
5708 031210 000200
5709
5710
5711
5712
5713
5714
5715
5716
5717 031212 004737 023460
5718 031216 113701 023560
5719 031222 113702 023561
5720 031226 013703 023556
5721 031232 105701
5722 031234 002403
5723 031236 123701 001630
5724 031242 103003
5725 031244 000241
5726 031246 106001
5727 031250 000772
5728 031252 105702
5729 031254 002403
5730 031256 122702 000023
5731 031262 003003
5732 031264 000241
5733 031266 106002
5734 031270 000772
5735 031272 023703 001510
5736 031276 003413
5737 031300 000241
5738 031302 006003
5739 031304 005503
5740 031306 001371
5741 031310 010103
5742 031312 000303
5743 031314 060203
5744 031316 005203
5745 031320 003364
5746 031322 005403
5747 031324 000762
5748 031326 023703 001512
5749 031332 002003
5750 031334 000241
5751 031336 006003
5752 031340 000772
5753 031342 023703 001510
5754 031346 003403
    
```

```

RANPAT: MOV R2,-(SP) ;SAVE R2
        MOV #256./2.,R2 ;GENERATE 256 WORDS
        BR 2$
1$: JSR PC,#SRAND ;GENERATE A RANDOM NUMBER
2$: MOV #LONUM,(R1)+ ;PUT LOW WORD IN BUFFER
    MOV #SHINUM,(R1)+ ;PUT HIGH WORD IN BUFFER
    DEC R2 ;DONE?
    BGT 1$ ;NO--BRANCH
    MOV (SP)+,R2 ;RESTORE R2
    RTS R0 ;EXIT
    
```

: THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR ADDRESSES AND SAVES THEM IN THE DPB (DTADPB+10 AND DTADPB+12).
 : NOTE: THIS ROUTINE DESTROYS R1-R3

```

: CALL
: JSR R0,#RANADR
: RETURN
    
```

```

RANADR: JSR PC,#SRAND ;GENERATE A RANDOM NUMBER
        MOV# #LONUM,R1 ;FORM SECTOR IN R1
        MOV# #LONUM+1,R2 ;FORM TRACK IN R2
        MOV #SHINUM,R3 ;FORM CYLINDER IN R3
        TSTB R1 ;ENSURE THE SECTOR IS BETWEEN 0 AND 21
        BLT 2$
1$: CMP# #LIMIT+22,R1 ;CHECK MAXIMUM SECTOR ADDRESS
    BHS 3$
2$: CLC
    RORB R1
    BR 1$
3$: TSTB R2 ;ENSURE THE TRACK IS BETWEEN 0 AND 18
    BLT 5$
4$: CMP# #19.,R2
    BGT 6$
5$: CLC
    RORB R2
    BR 4$
6$: CMP #FC,R3 ;ENSURE THE CYLINDER IS BETWEEN FC AND LC
    BLE 7$
    ROR R3
    ADC R3
    BNE 6$
    MOV R1,R3
    SWAB R3
    ADD R2,R3
    INC R3
    BGT 6$
    NEG R3
    BR 6$
7$: CMP #LC,R3
    BGE 8$
    CLC
    ROR R3
    BR 7$
8$: CMP #FC,R3
    BLE 9$
    
```


5755 031350 005203
5756 031352 000303
5757 031354 000764
5758 031356 110137 004174
5759 031362 110237 004175
5760 031366 010337 004176
5761 031372 000200
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771 031374 032777 000200 147536
5772 031402 001430
5773 031404 104401 031412
5774 031410 000410
5775
5776 031432
5777 031432 012703 043020
5778 031436 013704 001220
5779 031442 004037 031466
5780 031446 000771
5781 031450 000240
5782 031452 013737 001220 001222
5783 031460 010437 001220
5784 031464 000207
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801 031466 010337 031474
5802 031472 104401
5803 031474 000000
5804 031476 010446
5805 031500 104402
5806 031502 104401 043044
5807 031506 104411
5808 031510 012601
5809 031512 004037 033736
5810 031516 031472

```
INC R3
SMAB R3
BR 76
9S: MOV B R1, @DTADPB+10 ;SAVE SECTOR ADDRESS
MOV B R2, @DTADPB+11 ;SAVE TRACK ADDRESS
MOV B R3, @DTADPB+12 ;SAVE CYLINDER ADDRESS
RTS R0 ;RETURN

: THIS ROUTINE IS USED TO INPUT THE "CONTROL SWITCHES".
: IF SWR<07>=1 THE PRESENT SETTING WILL BE TYPED AND THE NEW
: SETTING IS READ AND STORED.
: NOTE: THIS ROUTINE DESTROYS R3 AND R4
: CALL
: JSR PC, @GETSWR
: RETURN ;(C.SWR)=DESIRED CONTROL SWITCHES

GETSWR: BIT @SW07, @SWR ;READ CONTROL SWITCHES?
BEQ 25 ;NO--BRANCH
TYPE 655 ;TYPE ASCIZ STRING
BR 645 ;GET OVER THE ASCIZ
: 655: .ASCIZ <CR><LF>/SET SWR<07>=0/
: 645:
1S: MOV @MSG.CS, R3 ;"CONTROL SWITCHES="
MOV @C.SWR, R4 ;PRESENT CONTROL SWITCH SETTINGS
JSR R0, @GETNUM ;GET THE NEW SWITCH SETTINGS
BR 1S ;COMMA
NOP ;PERIOD
MOV C.SWR, SAVCSW ;SAVE PREVIOUS VALUE
MOV R4, @C.SWR ;DOUBLE PERIOD--SAVE NEW SWITCH SETTING
2S: RTS PC ;RETURN FROM CALL

: THIS ROUTINE WILL TYPE AN ASCIZ MESSAGE AND THEN
: INPUT AN ASCIZ STRING AND CHANGE THE STRING TO OCTAL
: IF REQUIRED.
: NOTE: THIS ROUTINE DESTROYS R1
: CALL
: MOV @ADR, R3 ;ADDRESS OF ASCIZ MESSAGE
: MOV @NUM, R4 ;OCTAL NUMBER
: JSR R0, @GETNUM
: RETURN1 ;INPUT TERMINATED WITH A COMMA
: RETURN2 ;WITH A PERIOD
: RETURN3 ;WITH A DOUBLE PERIOD
: ;R4=INPUT NUMBER AND
: ;R2=R4*32 FOR ALL
: ;THREE RETURNS

GETNUM: MOV R3, 25 ;SAVE MESSAGE POINTER
1S: TYPE ;TYPE THE MESSAGE
2S: .WORD 0 ;MESSAGE POINTER GOES HERE
MOV R4, -(SP) ;SAVE R4 FOR TYPEOUT
TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE ,SLASH /
RDLIN ;READ AN ASCIZ STRING
MOV (SP)+, R1 ;ADDRESS OF FIRST CHARACTER
JSR R0, @CK.CHAR ;CHECK ONE CHARACTER
1S ;ILLEGAL CHARACTER
```



```

5811 031520 031472          1$          :CARRIAGE RETURN
5812 031522 031534          3$          :
5813 031524 031560          7$          :
5814 031526 031566          8$          :
5815 031530 031532          11$         :DIGIT 0-9
5816 031532 005301          11$: DEC      R1      :DECREMENT THE INPUT POINTER
5817 031534
5818 031534 004037 034176    3$:          :
5819 031540 031472          JSR      RD, @CK.NUM :CHECK THE NUMBER
5820 031542 031554          1$          :ILLEGAL INPUT
5821 031544 031552          6$          :TERMINATED WITH A "." OR "CR"
5822 031546 031550          5$          :TERMINATED WITH A ",'"
5823 031550 005720          4$: TST      (R0)+   :TERMINATED WITH A "..."
5824 031552 005720          5$: TST      (R0)+   :DOUBLE PERIOD
5825 031554 010204          6$: MOV      R2, R4   :SINGLE PERIOD
5826 031556 000414          BR       10$       :COMMA--SAVE INPUT NUMBER
5827 031560 105711          7$: TSTB    (R1)     :GO TO EXIT
5828 031562 001343          BNE      1$        :TERMINATOR AFTER A COMMA?
5829 031564 000411          BR       10$       :NO--LOOP
5830 031566 105711          8$: TSTB    (R1)     :YES--EXIT
5831 031570 001406          BEQ      9$        :TERMINATOR AFTER A PERIOD?
5832 031572 122721 000056  CMPB    #'.,(R1)+ :YES--EXIT
5833 031576 001335          BNE      1$        :NO--DOUBLE PERIOD?
5834 031600 105711          TSTB    (R1)     :NO--LOOP
5835 031602 001333          BNE      1$        :YES--TERMINATOR?
5836 031604 005720          TST      (R0)+   :NO--LOOP
5837 031606 005720          9$: TST      (R0)+   :DOUBLE PERIOD
5838 031610 010402          10$: MOV     R4, R2   :PERIOD
5839 031612 000302          SWAB    R2        :COMMA--POSITION THE
5840 031614 006202          ASR     R2        :NUMBER IN CASE IT
5841 031616 006202          ASR     R2        :IS THE PRIORITY LEVEL
5842 031620 006202          ASR     R2
5843 031622 000200          RTS      R0      :EXIT
5844
5845 :THIS ROUTINE IS USED TO CHANGE OR MODIFY
5846 :THE TEST PARAMETERS. IT GIVES THE OPERATOR
5847 :THE CAPABILITY OF SPECIFYING WHICH DRIVES TO TEST, WHICH
5848 :TESTS TO RUN AND HOW MANY TIMES TO
5849 :REPEAT EACH TEST
5850
5851 031624 104412 001232    GT.PRM: SAVREG    :SAVE R0 - R5
5852 031626 005037 031640  GT.PR1: CLR      DRVSEL :NO DRIVE SELECTED
5853 031632 104401 031640    TYPE    65$       :TYPE ASCIZ STRING
5854 031636 000406          BR       64$       :GET OVER THE ASCIZ
5855
5856 031654          ::65$: .ASCIZ  <CR><LF>/DRIVE(S)=/
5857 031654 104411          64$:          :
5858 031656 012601          RDLIN    :READ TTY
5859 031660 004037 033736  MOV     (SP)+, R1  :ADDRESS OF ASCIZ STRING
5860 031664 031626          JSR     RD, @CK.CHR :CHECK ONE CHARACTER
5861 031666 031626          GT.PR1   :ILLEGAL CHARACTER
5862 031670 031626          GT.PR1   :CARRIAGE RETURN
5863 031672 031626          GT.PR1   :
5864 031674 031626          GT.PR1   :
5865 031676 031700          1$: IS      :DIGIT 0-9
5866 031700 005301          DEC      R1

```



```

5867 031702          25:
5868 031702 012702 000007      MOV      #7,R2      :UPPER LIMIT OF INPUT
5869 031706 004037 034012      JSR      RD,#CK.DIG :CHECK THE DIGIT(S)
5870 031712 031626          GT.PRI1          :ILLEGAL INPUT
5871 031714 031626          GT.PRI1          :INPUT TO LARGE
5872 031716 031724          35:          :TERMINATED WITH A "." OR "CR"
5873 031720 031750          45:          :TERMINATED WITH A "."
5874 031722 031750          45:          :TERMINATED WITH A "."
5875 031724 156237 034452 001232 35:  BISB    ATABIT(R2),DRVSEL :SET THE DRIVE SELECTED BIT
5876 031732 105741          TSTB    -(R1)       :WAS THE LINE TERMINATED?
5877 031734 001362          BNE     25         :NO-GET THE NEXT DRIVE
5878 031736 005037 001234      CLR     #TSTNMS    :DESELECT ALL TESTS
5879 031742 005037 001236      CLR     TSTNMS+2
5880 031746 000405          BR      GTTST1     :YES--SELECT TEST
5881 031750 156237 034452 001232 45:  BISB    ATABIT(R2),DRVSEL :SET THE SELECTED DRIVE BITS
5882 031756 104413          GT.PR2: RESREG    :RESTORE RD - R5
5883 031760 000207          RTS     PC        :EXIT
5884
5885 031762          GTTST1:
5886 031762 104401 031770      TYPE   #65$      ;;TYPE ASCIZ STRING
5887 031766 000403          BR     #64$     ;;GET OVER THE ASCIZ
5888
5889 031776          ;;65$: .ASCIZ  /TEST=/
5890 031776 104411          64$:          RDLIN          :READ AN ASCIZ STRING
5891 032000 012601          MOV     (SP)+,R1  :POINTER TO R1
5892 032002 122711 000056      CMPB   #'.,(R1)  :DOUBLE PERIOD?
5893 032006 001007          BNE     15       :NO--BRANCH
5894 032010 122761 000056 000001  CMPB   #'.,1(R1)
5895 032016 001003          BNE     15
5896 032020 105761 000002      TSTB   2(R1)     : "CR"?
5897 032024 001754          BEQ    GT.PR2    : YES--EXIT
5898 032026 005037 001234      15:    CLR     TSTNMS  : NO TEST SELECTED
5899 032032 005037 001236      CLR     TSTNMS+2
5900 032036 005037 001240      CLR     OPNFLG   : NO TESTS TO BE OPENED
5901 032042 005037 001242      CLR     OPNFLG+2
5902 032046 121127 000123      GTTST2: CMPB   (R1),#'S  : ALL SEEK TESTS?
5903 032052 001004          BNE     15       : NO--BRANCH
5904 032054 052737 000777 001234  BIS    #777,TSTNMS : YES--SELECT TESTS 0-10
5905 032062 000552          BR      GTTST3
5906 032064 121127 000124      15:    CMPB   (R1),#'T  : ALL TIMING TESTS?
5907 032070 001004          BNE     25       : NO--BRANCH
5908 032072 052737 036000 001234  BIS    #36000,TSTNMS : YES--SELECT TESTS 12-15
5909 032100 000543          BR      GTTST3
5910 032102 121127 000101      25:    CMPB   (R1),#'A  : ALL ADDRESSING TESTS?
5911 032106 001004          BNE     35       : NO--BRANCH
5912 032110 052737 140000 001234  BIS    #140000,TSTNMS : YES--SELECT TESTS 16 & 17
5913 032116 000534          BR      GTTST3
5914 032120 121127 000104      35:    CMPB   (R1),#'D  : DATA TEST?
5915 032124 001004          BNE     45       : NO--BRANCH
5916 032126 052737 000001 001236  BIS    #1,TSTNMS+2 : YES--SELECT TEST 20
5917 032134 000525          BR      GTTST3
5918 032136 121127 000105      45:    CMPB   (R1),#'E  : EXERCISER TEST?
5919 032142 001004          BNE     55       : NO--BRANCH
5920 032144 052737 000002 001236  BIS    #2,TSTNMS+2 : YES--SELECT TEST 21
5921 032152 000516          BR      GTTST3
5922 032154 004037 033662      55:    JSR     RD,#CK.OCT : OCTAL DIGIT?
    
```


5923	032160	000514		BR	GTTST4	:NO--BRANCH
5924	032162	010205		MOV	R2,R5	:YES--SAVE IT
5925	032164	005201		INC	R1	:MOVE TO NEXT CHARACTER
5926	032166	004037	033662	JSR	RO, @CK.OCT	:OCTAL DIGIT
5927	032172	000405		BR	6\$:NO--BRANCH
5928	032174	005201		INC	R1	:MOVE TO NEXT CHARACTER
5929	032176	006305		ASL	R5	:SCALE HIGH DIGIT
5930	032200	006305		ASL	R5	
5931	032202	006305		ASL	R5	
5932	032204	060502		ADD	R5,R2	:COMBINE HIGH & LOW DIGITS
5933	032206	020227	000022	6\$: CMP	R2, @STN-1	:VALID TEST NUMBER?
5934	032212	003263		BGT	GTTST1	:NO--BRANCH
5935	032214	010237	032406	MOV	R2, 9\$:SAVE THE TEST NUMBER
5936	032220	010204		MOV	R2,R4	:CONVERT TEST NUMBER INTO AN INDEX
5937	032222	042704	000017	BIC	@17,R4	:CLEAR UNWANTED BITS
5938	032226	006204		ASR	R4	:SHIFT THE BITS
5939	032230	006204		ASR	R4	:SHIFT THE BITS
5940	032232	006204		ASR	R4	:SHIFT THE BITS
5941	032234	006302		ASL	R2	
5942	032236	056264	001424 001234	BIS	BITS(R2), TSTNMS(R4)	:SELECT TEST
5943	032244	121127	000055	CMPB	(R1), @'-	:TEST STRING?
5944	032250	001060		BNE	GTTST4	:NO--BRANCH
5945	032252	005201		INC	R1	:YES--MOVE TO NEXT CHARACTER
5946	032254	004037	033662	JSR	RO, @CK.OCT	:OCTAL DIGIT?
5947	032260	000640		BR	GTTST1	:NO--BRANCH
5948	032262	010205		MOV	R2,R5	:YES--SAVE IT
5949	032264	005201		INC	R1	:MOVE TO NEXT CHARACTER
5950	032266	004037	033662	JSR	RO, @CK.OCT	:OCTAL DIGIT?
5951	032272	000405		BR	7\$:NO--BRANCH
5952	032274	005201		INC	R1	:YES--MOVE TO NEXT CHARACTER
5953	032276	006305		ASL	R5	:SCALE HIGH DIGIT
5954	032300	006305		ASL	R5	
5955	032302	006305		ASL	R5	
5956	032304	060502		ADD	R5,R2	:COMBINE HIGH & LOW DIGIT
5957	032306	020227	000022	7\$: CMP	R2, @STN-1	:VALID TEST NUMBER?
5958	032312	003223		BGT	GTTST1	:NO--BRANCH
5959	032314	023702	032406	CMP	9\$, R2	:IS THE FIRST NUMBER OF THE
5960						:STRING SMALLER THAN THE LAST?
5961	032320	002220		BGE	GTTST1	:NO--BRANCH
5962	032322	010246		MOV	R2, -(SP)	:SAVE ENDING TEST NUMBER
5963	032324	013702	032406	MOV	9\$, R2	:GET STARTING TEST NUMBER
5964	032330	012637	032406	MOV	(SP)+, 9\$:STORE ENDING TEST NUMBER
5965	032334	006337	032406	ASL	9\$:SHIFT ENDING TEST NUMBER
5966	032340	006302		ASL	R2	:SHIFT TEST NUMBER
5967	032342	010204		MOV	R2,R4	:COPY TEST NUMBER INTO R4
5968	032344	042704	000037	8\$: BIC	@37,R4	:CLEAR LOWER BITS
5969	032350	006204		ASR	R4	:SHIFT THE TEST NUMBER
5970	032352	006204		ASR	R4	:SHIFT THE TEST NUMBER
5971	032354	006204		ASR	R4	:SHIFT THE TEST NUMBER
5972	032356	006204		ASR	R4	:SHIFT THE TEST NUMBER
5973	032360	056264	001424 001234	BIS	BITS(R2), TSTNMS(R4)	:SELECT THE TEST
5974	032366	062702	000002	ADD	@2,R2	:INCREMENT THE TEST NUMBER
5975	032372	020237	032406	CMP	R2, 9\$:SEE IF FINISHED
5976	032376	101761		BLOS	8\$:BR IF NOT
5977	032400	162702	000002	SUB	@2,R2	:CORRECT TEST NUMBER
5978	032404	000402		BR	GTTST4	:CONTINUE


```

5979 032406 000000          9S: .WORD 0 ;STORE TEST NUMBER HERE
5980 032410 005201          GTTST3: INC R1 ;MOVE TO NEXT CHARACTER
5981 032412 121127 000056          GTTST4: CMPB (R1),#' ;"PERIOD"?
5982 032416 001441          BEQ GTTST5 ;YES--BRANCH
5983 032420 005737 001234          TST TSTNMS ;ANY TEST SELECTED THIS CYCLE?
5984 032424 001005          BNE IS ;BR IF YES
5985 032426 005737 001236          TST TSTNMS+2 ;ANY TEST SELECTED THIS CYCLE ?
5986 032432 001002          BNE IS ;BR IF YES
5987 032434 000137 031762          JMP GTTST1 ;NO
5988 032440 121127 000057          1S: CMPB (R1),#'/ ;"OPEN"?
5989 032444 001004          BNE 2S ;NO--BRANCH
5990 032446 056264 001424 001240          BIS BITS(R2), OPNFLG(R4) ;YES--SET BITS FOR TEST TO OPEN
5991 032454 000405          BR 3S
5992 032456 121127 000054          2S: CMPB (R1),#', ;"COMMA"?
5993 032462 001402          BEQ 3S ;BR IF YES
5994 032464 000137 031762          JMP GTTST1 ;NO
5995 032470 005201          3S: INC R1 ;MOVE TO NEXT CHARACTER
5996 032472 105711          TSTB (R1) ;"CR"?
5997 032474 001402          BEQ 4S ;BR IF 'CR'
5998 032476 000137 032046          JMP GTTST2 ;NO--GO GET NEXT CHARACTER
5999 032502 005737 001240          4S: TST OPNFLG ;ANY TESTS TO OPEN ?
6000 032506 001042          BNE OPNTST ;BR IF YES
6001 032510 005737 001242          TST OPNFLG+2 ;ANY TESTS TO OPEN ?
6002 032514 001037          BNE OPNTST ;BR IF YES
6003 032516 000137 031762          JMP GTTST1 ;NO--START AGAIN
6004 032522 005201          GTTST5: INC R1 ;MOVE TO NEXT CHARACTER
6005 032524 121127 000056          CMPB (R1),#' ;"PERIOD"?
6006 032530 001414          BEQ GTTST6 ;YES--BRANCH
6007 032532 105711          TSTB (R1) ;"CR"?
6008 032534 001402          BEQ IS ;YES--BRANCH
6009 032536 000137 031762          JMP GTTST1
6010 032542 005737 001240          1S: TST OPNFLG ;ANY TESTS TO OPEN ?
6011 032546 001022          BNE OPNTST ;BR IF YES
6012 032550 005737 001242          TST OPNFLG+2 ;ANY TESTS TO OPEN ?
6013 032554 001017          BNE OPNTST ;BR IF YES
6014 032556 000137 031756          JMP GT.PR2 ;NO--GO START TESTING
6015 032562 005201          GTTST6: INC R1 ;MOVE TO NEXT CHARACTER
6016 032564 105711          TSTB (R1) ;"CR"?
6017 032566 001402          BEQ IS ;YES--BRANCH
6018 032570 000137 031762          JMP GTTST1 ;NO--GO ASK FOR TEST
6019 032574 005737 001240          1S: TST OPNFLG ;ANY TESTS TO OPEN ?
6020 032600 001005          BNE OPNTST ;BR IF YES
6021 032602 005737 001242          TST OPNFLG+2 ;ANY TESTS TO OPEN ?
6022 032606 001002          BNE OPNTST ;BR IF YES
6023 032610 000137 031756          JMP GT.PR2 ;NO--GO START TESTING
6024
6025
6026
6027 032614 104412          ;OPEN THE SELECTED TEST FOR CHANGES
6028 032616 005027          OPNTST: SAVREG
6029 032620 000000          CLR (PC)+
6030 032622 000411          OPN.CT: .WORD 0 ;SAVE R0 - R5
6031 032624 005237 032620          BR OPN.2 ;START WITH TEST 0
6032 032630 022737 000022 032620          OPN.1: INC OPN.CT ;COUNT STORED HERE
6033 032636 002003          CMP #STN-1,OPN.CT ;SKIP THE INCREMENT
6034 032640 104413          BGE OPN.2 ;MOVE TO THE NEXT TEST
          RESREG ;TEST NUMBER TOO BIG?
          ;NO--OPEN THE NEXT TEST
          ;RESTORE R0 - R5

```



```

6035 032642 000137 031762      JMP      GTTST1      ;YES--GO ASK FOR MORE TESTS
6036 032646 013705 032620      OPN.2: MOV      OPN.CT,R5 ;SETUP TO USE THE
6037 032652 006305                ASL      R5          ;TEST NUMBER AS AN INDEX
6038 032654 013703 032620      MOV      OPN.CT,R3  ;GET INDEX
6039 032660 042703 000017      BIC      #17,R3     ;CLEAR LOWER TEST BITS
6040 032664 006203                ASR      R3          ;SHIFT TEST NUMBER
6041 032666 006203                ASR      R3          ;SHIFT TEST NUMBER
6042 032670 006203                ASR      R3          ;SHIFT TEST NUMBER
6043 032672 036563 001424 001240  BIT      BITS(R5),OPNFLG(R3) ;OPEN THIS TEST?
6044 032700 001751                BEQ      OPN.1      ;NO--MOVE TO NEXT TEST
6045 032702 104401 032710      TYPE    '65$'      ;TYPE ASCIZ STRING
6046 032706 000404                BR       64$        ;GET OVER THE ASCIZ
6047                                ;:65$:
6048 032720                                64$: .ASCIZ  / TEST /
6049 032720 013746 032620      MOV      OPN.CT,-(SP) ;SAVE OPN.CT FOR TYPEOUT
6050                                ;:TEST NUMBER
6051 032724 104403                TYPOS   2           ;GO TYPE--OCTAL ASCII
6052 032726 002                .BYTE  2           ;TYPE 2 DIGIT(S)
6053 032727 000                .BYTE  0           ;SUPPRESS LEADING ZEROS
6054 032730 104401 001215      TYPE    'SCRLF'    ;TYPE "CR" & "LF"
6055 032734 016500 001536      MOV      PRMPT(R5),R0 ;PICKUP PARAMETER POINTER
6056 032740 011046                MOV      (R0),-(SP) ;SAVE THE VARIABLE INDICATOR
6057 032742 012702 001504      MOV      #PRM,R2    ;FIRST ADDRESS OF TABLE
6058 032746 000405                BR       2$         ;
6059 032750 006216      1$: ASR      (SP)      ;CHECK FOR A VARIABLE
6060 032752 103403                BCS     2$         ;GO MOVE THIS ONE
6061 032754 001404                BEQ     OPNPRM    ;DONE
6062 032756 005722                TST    (R2)+     ;BUMP THE POINTER
6063 032760 000773                BR     1$         ;
6064 032762 012022      2$: MOV      (R0)+,(R2)+ ;MOVE THIS VARIABLE INTO THE
6065 032764 000771                BR     1$         ;COMMON AREA
6066 032766 013716 001504  OPNPRM: MOV    @#PRM,(SP) ;GET THE VARIABLE INDICATOR
6067 032772 005004                CLR    R4        ;ZERO THE INDEX
6068 032774 006216      1$: ASR      (SP)      ;CHECK FOR A VARIABLE
6069 032776 103403                BCS     3$         ;GO GET IT
6070 033000 001772                BEQ     OPNPRM    ;OUT OF VARIABLES
6071 033002 005724      2$: TST    (R4)+     ;UPDATE THE INDEX
6072 033004 000773                BR     1$         ;
6073 033006 005764 001606      3$: TST    PRMLT(R4) ;IS THE MAX. MAGNITUDE NEG?
6074 033012 100466                BMI    OPNPAT    ;YES--THEN IT IS THE PATTERN
6075 033014 104401 044010      TYPE    'MSG.SP'  ;TYPE SPACES
6076 033020 016437 001636 033030  MOV      PRMSG(R4),4$ ;TYPE THE NAME OF THIS VARIABLE
6077 033026 104401                TYPE    0         ;
6078 033030 000000      4$: .WORD  0         ;TYPE "="
6079 033032 104401 043016      TYPE    'MSG.EQ' ;PUT RPT(R4) ON THE STACK
6080 033036 016446 001506      MOV      RPT(R4),-(SP) ;CHANGE RPT(R4) TO DECIMAL ASCIZ
6081 033042 004737 023170      JSR     PC,@#SSB2D ;TYPE WITHOUT LEADING ZEROS
6082 033046 004737 023420      JSR     PC,@#SSUPRS ;
6083 033052 104401 043044      TYPE    'SLASH'  ;
6084 033056 104411                RDLIN  ;
6085 033060 012601                MOV      (SP)+,R1  ;READ AN ASCIZ STRING
6086 033062 004037 033736      JSR     R0,@#CK.CHR ;CHECK ONE CHARACTER
6087 033066 033006                3$          ;ILLEGAL CHARACTER
6088 033070 033002                2$          ;CARRIAGE RETURN
6089 033072 033140                1$          ;
6090 033074 033102                5$          ;

```


6091	033076	033110		6S				
6092	033100	033136		7S				
6093	033102	105711		5S:	TSTB	(R1)		DIGIT 0-9
6094	033104	001340			BNE	3S		"CR"?
6095	033106	000735			BR	2S		NO--STAY ON THIS VARIABLE
6096	033110	105711		6S:	TSTB	(R1)		YES--MOVE TO NEXT VARIABLE
6097	033112	001002			BNE	64S		IS THERE A "CR" AFTER THE PERIOD?
6098	033114	000137	033530		JMP	OPN.N2		NO
6099	033120	122721	000056	64S:	CMPB	#'.,(R1)+		YES--GO CLOSE THIS TEST
6100	033124	001330			BNE	3S		DOUBLE PERIOD?
6101	033126	105711			TSTB	(R1)		NO--GO ASK FOR THIS VARIABLE
6102	033130	001326			BNE	3S		YES--IS A "CR" AFTER THE DOUBLE PERIOD?
6103	033132	000137	033546		JMP	OPN.X2		NO--ASK FOR THIS VARIABLE AGAIN
6104	033136	005301		7S:	DEC	R1		YES--CLOSE ALL TEST
6105	033140			8S:				BACK THE POINTER UP BY ONE
6106	033140	016402	001606		MOV	PRMLMT(R4),R2		UPPER LIMIT OF INPUT
6107	033144	004037	034012		JSR	RD,#CK.DIG		CHECK THE DIGIT(S)
6108	033150	033006			3S			ILLEGAL INPUT
6109	033152	033006			3S			INPUT TO LARGE
6110	033154	033162			9S			TERMINATED WITH A ".," OR "CR"
6111	033156	033524			OPN.N1			TERMINATED WITH A ".:"
6112	033160	033542			OPN.X1			TERMINATED WITH A "..."
6113	033162	010264	001506	9S:	MOV	R2,RPT(R4)		SAVE THIS VARIABLE
6114	033166	000705			BR	2S		MOVE TO NEXT VARIABLE
6115	033170	104401	044010	OPNPAT:	TYPE	,MSG.SP		TYPE SPACES
6116	033174	104401	043012		TYPE	,MSG.PAT		TYPE "PAT"
6117	033200	104401	043016		TYPE	,MSG.EQ		TYPE "="
6118	033204	016446	001506		MOV	RPT(R4),-(SP)		SAVE RPT(R4) FOR TYPEOUT
6119	033210	104402			TYPOC			GO TYPE--OCTAL ASCII(ALL DIGITS)
6120	033212	104401	044011		TYPE	,MSG.SP+1		TYPE ONE SPACE
6121	033216	104411			RDLIN			READ ASCII STRING
6122	033220	012601			MOV	(SP)+,R1		PICKUP POINTER
6123	033222	004037	033736		JSR	RD,#CK.CHR		CHECK ONE CHARACTER
6124	033226	033170			OPNPAT			ILLEGAL CHARACTER
6125	033230	032766			OPNPRM			CARRIAGE RETURN
6126	033232	033264			3S			"/"
6127	033234	032766			OPNPRM			":"
6128	033236	033242			1S			"::"
6129	033240	033262			2S			DIGIT 0-9
6130	033242	105711		1S:	TSTB	(R1)		"CR" AFTER THE PERIOD?
6131	033244	001531			BEQ	OPN.N2		YES--GO CLOSE THIS TEST
6132	033246	122721	000056		CMPB	#'.,(R1)+		NO--PERIOD?
6133	033252	001346			BNE	OPNPAT		NO--LOOP
6134	033254	105711			TSTB	(R1)		"CR" AFTER A DOUBLE PERIOD?
6135	033256	001533			BEQ	OPN.X2		YES--GO START TESTING
6136	033260	000743			BR	OPNPAT		NO--LOOP
6137	033262	005301		2S:	DEC	R1		BACKUP THE ASCII POINTER
6138	033264			3S:				
6139	033264	004037	034176		JSR	RD,#CK.NUM		CHECK THE NUMBER
6140	033270	033170			OPNPAT			ILLEGAL INPUT
6141	033272	033300			4S			TERMINATED WITH A ".," OR "CR"
6142	033274	033524			OPN.N1			TERMINATED WITH A ".:"
6143	033276	033542			OPN.X1			TERMINATED WITH A "..."
6144	033300	010264	001506	4S:	MOV	R2,RPT(R4)		SAVE THE INPUT NUMBER
6145	033304	006002			ROR	R2		OPEN PATTERN 0?
6146	033306	103227			BCC	OPNPRM		NO--START AT BEGINNING OF PARAMETER TABLE

6147	033307	104412			OPNWD5: SAVREG	:SAVE R0 - R5
6148	033312	005000			CLR R0	:START WITH WORD 0
6149	033314	012704	003104		MOV #PATO,R4	
6150	033320			15:		
6151	033320	104401	033326		TYPE 655	::TYPE ASCIZ STRING
6152	033324	000403			BR 645	::GET OVER THE ASCIZ
6153				::655:	.ASCIZ / WD/	
6154	033334			645:		
6155	033334	010046			MOV R0,-(SP)	:PUT R0 ON THE STACK
6156	033336	004737	023170		JSR PC,@#5SB2D	:CHANGE R0 TO DECIMAL ASCIZ
6157	033342	004737	023420		JSR PC,@#5SUPRS	:TYPE WITHOUT LEADING ZEROS
6158	033346	104401	043016		TYPE MSG.EQ	:TYPE "="
6159	033352	011446			MOV (R4),-(SP)	:SAVE (R4) FOR TYPEOUT
6160	033354	104402			TYPOC	:GO TYPE--OCTAL ASCII(ALL DIGITS)
6161	033356	104411			RDLIN	:READ ASCIZ STRING
6162	033360	012601			MOV (SP)+,R1	:PICKUP THE POINTER
6163	033362	004037	033736		JSR R0,@#CK.CHR	:CHECK ONE CHARACTER
6164	033366	033320			15	:ILLEGAL CHARACTER
6165	033370	033422			45	:CARRIAGE RETURN
6166	033372	033404			25	
6167	033374	033422			45	
6168	033376	033436			55	
6169	033400	033402			105	:DIGIT 0-9
6170	033402	005301		105:	DEC R1	:BACKUP THE ASCII POINTER
6171	033404			25:		
6172	033404	004037	034176		JSR R0,@#CK.NUM	:CHECK THE NUMBER
6173	033410	033320			15	:ILLEGAL INPUT
6174	033412	033420			35	:TERMINATED WITH A "." OR "CR"
6175	033414	033456			65	:TERMINATED WITH A "..."
6176	033416	033470			85	:TERMINATED WITH A "..."
6177	033420	010214		35:	MOV R2,(R4)	:SAVE THE INPUT
6178	033422	005724		45:	TST (R4)+	:MOVE TO NEXT WORD
6179	033424	005200			INC R0	:INCREMENT THE COUNT
6180	033426	022700	000020		CMP #16.,R0	:COUNT TO LARGE?
6181	033432	003332			BGT 15	:NO--BRANCH
6182	033434	000726			BR OPNWD5	:YES--BRANCH
6183	033436	105711		55:	TSTB (R1)	: "CR" AFTER THE PERIOD?
6184	033440	001407			BEQ 75	:YES--GO CLOSE THIS TEST
6185	033442	122721	000056		CMPB #'.,(R1)+	:NO--PERIOD?
6186	033446	001324			BNE 15	:NO--BRANCH ILLEGAL INPUT STRING
6187	033450	105711			TSTB (R1)	: "CR" AFTER THE "PERIOD-PERIOD"?
6188	033452	001407			BEQ 95	:YES--GO START TESTING
6189	033454	000721			BR 15	:NO--LOOP
6190	033456	010224		65:	MOV R2,(R4)+	:SAVE THE INPUT
6191	033460	004737	033502	75:	JSR PC,@#CLSWDS	:CLOSE THE DATA PATTERN
6192	033464	104413			RESREG	:RESTORE R0 - R5
6193	033466	000420			BR OPN.N2	:MOVE TO NEXT TEST
6194	033470	010224		85:	MOV R2,(R4)+	:SAVE THE INPUT
6195	033472	004737	033502	95:	JSR PC,@#CLSWDS	:CLOSE THE DATA PATTERN
6196	033476	104413			RESREG	:RESTORE R0 - R5
6197	033500	000422			BR OPN.X2	:START TESTING
6198	033502	012701	003104	CLSWS:	MOV #PATO,R1	:FIRST ADDRESS OF DATA PATTERN
6199	033506	005200		15:	INC R0	:COUNT THE LAST WORD THAT WAS STORED
6200	033510	022700	000017		CMP #15.,R0	:END OF TABLE
6201	033514	002402			BLT 25	:YES--EXIT
6202	033516	012124			MOV (R1)+,(R4)+	:COPY

010

6203	033520	000772		BR	1\$: LOOP
6204	033522	000207		2\$: RTS	PC	: RETURN
6205	033524	010264	001506	OPN.N1: MOV	R2,RPT(R4)	: SAVE THIS VARIABLE
6206	033530	005726		OPN.N2: TST	(SP)+	: CLEAN OFF THE STACK
6207	033532	004737	033602	JSR	PC,CLOSE	: CLOSE THIS TEST
6208	033536	000137	032624	JMP	OPN.1	: GO OPEN THE NEXT TEST
6209	033542	010264	001506	OPN.X1: MOV	R2,RPT(R4)	: SAVE THIS VARIABLE
6210	033546	005726		OPN.X2: TST	(SP)+	: CLEAN OFF THE STACK
6211	033550	004737	033602	1\$: JSR	PC,CLOSE	: CLOSE THIS TEST
6212	033554	005725		2\$: TST	(RS)+	: UPDATE THE INDEX
6213	033556	020527	000034	CMP	RS,#16*2	: INDEX TO BIG?
6214	033562	002403		BLT	3\$: NO--BRANCH
6215	033564	104413		RESREG		: RESTORE R0 - R5
6216	033566	000137	031756	JMP	GT.PR2	: GO TO EXIT
6217	033572	036503	001424	3\$: BIT	BITS(R5),R3	: IS THIS TEST OPEN FOR CHANGE?
6218	033576	001364		BNE	1\$: YES--GO CLOSE IT
6219	033600	000765		BR	2\$: NO--MOVE TO NEXT TEST
6220	033602	104412		CLOSE: SAVREG		: SAVE R0 - R5
6221	033604	012700	001504	MOV	#PRM,R0	: "FROM" ADDRESS
6222	033610	016501	001536	MOV	PRMPT(R5),R1	: "TO" ADDRESS
6223	033614	012002		MOV	(R0)+,R2	: "FROM" INDICATOR
6224	033616	012103		MOV	(R1)+,R3	: "TO" INDICATOR
6225	033620	012704	000001	MOV	#1,R4	: TEST BIT START A "RPT"
6226	033624	030402		1\$: BIT	R4,R2	: PARAMETER TO BE MOVED?
6227	033626	001403		BEQ	2\$: NO--BRANCH
6228	033630	030403		BIT	R4,R3	: A PLACE TO PUT IT?
6229	033632	001404		BEQ	3\$: NO--BRANCH
6230	033634	011011		MOV	(R0),(R1)	: YES--MOVE "FROM" TO "TO"
6231	033636	030403		2\$: BIT	R4,R3	: "TO" PARAMETER?
6232	033640	001401		BEQ	3\$: NO--BRANCH
6233	033642	005721		TST	(R1)+	: YES--UPDATE THE POINTER
6234	033644	005720		3\$: TST	(R0)+	: UPDATE FROM POINTER
6235	033646	006304		ASL	R4	: POSITION THE TEST BIT
6236	033650	032704	002000	BIT	#BIT10,R4	: DONE?
6237	033654	001763		BEQ	1\$: NO--BRANCH
6238	033656	104413		RESREG		: RESTORE R0 - R5
6239	033660	000207		RTS	PC	: RETURN
6240				; THIS ROUTINE IS USED TO CHECK IF AN		
6241				; ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.		
6242				CALL		
6243				MOV	#ADR,R1	: ADDRESS OF ASCII CHARACTER
6244				JSR	R0,CHK.OCT	: CHECK THE CHARACTER
6245				RETURN1		: CHARACTER IS NOT BETWEEN 0-7
6246				RETURN2		: CHARACTER IS IN R2 AS A
6247						: OCTAL DIGIT
6248						
6249	033662	121127	000060	CK.OCT: CMPB	(R1),#0	: LESS THAN ZERO?
6250	033666	103407		BLO	1\$: YES -- BRANCH
6251	033670	121127	000067	CMPB	(R1),#7	: GREATER THAN SEVEN?
6252	033674	101004		BHI	1\$: YES -- BRANCH
6253	033676	111102		MOVB	(R1),R2	: GET THE CHARACTER
6254	033700	042702	177770	BIC	#7,R2	: STRIP AWAY THE ASCII
6255	033704	005720		TST	(R0)+	: ADJUST FOR RETURN
6256	033706	000200		1\$: RTS	R0	: RETURN
6257						
6258				; THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER		

6259
 6260
 6261
 6262
 6263
 6264
 6265
 6266
 6267 033710 121127 000060
 6268 033714 103407
 6269 033716 121127 000071
 6270 033722 101004
 6271 033724 111102
 6272 033726 042702 000060
 6273 033732 005720
 6274 033734 000200
 6275
 6276
 6277
 6278
 6279
 6280
 6281
 6282
 6283
 6284
 6285
 6286
 6287
 6288
 6289 033736 105711
 6290 033740 001420
 6291 033742 121127 000057
 6292 033746 001414
 6293 033750 121127 000054
 6294 033754 001410
 6295 033756 121127 000056
 6296 033762 001404
 6297 033764 004037 033710
 6298 033770 000406
 6299 033772 005720
 6300 033774 005720
 6301 033776 005720
 6302 034000 005720
 6303 034002 005720
 6304 034004 005201
 6305 034006 011000
 6306 034010 000200
 6307
 6308
 6309
 6310
 6311
 6312
 6313
 6314

```

:AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
CALL
      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
      JSR      RD,CHK.DEC   ;CHECK THE CHARACTER
      RETURN   ADR1        ;NOT BETWEEN 0 AND 9
      RETURN   ADR2        ;BETWEEN 0 AND 9
      RETURN   ADR3        ;R2 = DIGIT
    
```

```

CK.DEC: CMPB   (R1),#'0     ;LESS THAN ZERO?
        BLO    1$          ;YES -- BRANCH
        CMPB   (R1),#'9     ;GREATER THAN NINE?
        BHI    1$          ;YES -- BRANCH
        MOVB   (R1),R2      ;GET THE CHARACTER
        BIC    #'0,R2       ;STRIP AWAY THE ASCII
        TST    (R0)+        ;ADJUST FOR RETURN
1$:     RTS      RD         ;RETURN
    
```

: THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
 : DETERMINE WHAT IT IS.

```

CALL
      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
      JSR      RD,CHK.CHR   ;CHECK CHARACTER
      RETURN   ADR1        ;UNKNOWN CHARACTER
      RETURN   ADR2        ;CARRIAGE RETURN * (R1)=ADR+1
      RETURN   ADR3        ;SLASH * (R1)=ADR+1
      RETURN   ADR4        ;COMMA * (R1)=ADR+1
      RETURN   ADR5        ;PERIOD * (R1)=ADR+1
      RETURN   ADR6        ;DIGIT BETWEEN 0 AND 9.
      RETURN   ADR7        ;R2 = DIGIT * (R1)=ADR+1
    
```

```

CK.CHR: TSTB   (R1)        ;"CARRIAGE RETURN"?
        BEQ    4$          ;YES -- BRANCH
        CMPB   (R1),#'/'    ;"SLASH"?
        BEQ    3$          ;YES -- BRANCH
        CMPB   (R1),#','    ;"COMMA"?
        BEQ    2$          ;YES -- BRANCH
        CMPB   (R1),# '.'   ;"PERIOD"?
        BEQ    1$          ;YES -- BRANCH
        JSR    RD,CHK.DEC   ;"DIGIT"?
        BR     5$          ;NO -- BRANCH
        TST    (R0)+        ;DIGIT BETWEEN 0-9
1$:     TST    (R0)+        ;PERIOD
2$:     TST    (R0)+        ;COMMA
3$:     TST    (R0)+        ;SLASH
4$:     TST    (R0)+        ;CARRIAGE RETURN
        INC    R1           ;MOVE POINTER TO NEXT CHARACTER
5$:     MOV    (R0),R0      ;UNKNOWN CHARACTER
        RTS      RD         ;RETURN
    
```

: THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
 : CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.

```

CALL
      MOV      #ADR,R1      ;ADDRESS OF ASCII STRING
      MOV      #NUM,R2     ;MAX. MAGNITUDE OF INPUT NUMBER
      JSR      RD,CHK.DIG   ;CHECK DIGITS
      RETURN   ADR1        ;ILLEGAL CHARACTER -- R2=?
    
```



```

6315      :      RETURN  ADR2      : INPUT NUMBER TO LARGE -- R2=?
6316      :      RETURN  ADR3      : "COMMA" -- R2 = NUMBER
6317      :      RETURN  ADR4      : "PERIOD" -- R2 = NUMBER
6318      :      RETURN  ADR5      : "PERIOD-PERIOD" -- R2 = NUMBER
6319
6320      034012  010446      CK.DIG:  MOV      R4,-(SP)      : SAVE R4
6321      034014  010346      :      MOV      R3,-(SP)      : SAVE R3
6322      034016  010246      :      MOV      R2,-(SP)      : SAVE THE MAX. SIZE ON THE STACK
6323      034020  005002      :      CLR      R2            : START WITH 0
6324      034022  005003      :      CLR      R3
6325      034024  005004      :      CLR      R4
6326      034026  004037  033736  :      JSR      RD,@CK.CHR      : CHECK ONE CHARACTER
6327      034032  034162      :      BS      : ILLEGAL CHARACTER
6328      034034  034162      :      BS      : CARRIAGE RETURN
6329      034036  034162      :      BS      :
6330      034040  034162      :      BS      :
6331      034042  034162      :      BS      :
6332      034044  034046      :      IS      : DIGIT 0-9
6333      034046  006303      15:      ASL      R3            : 2
6334      034050  010346      :      MOV      R3,-(SP)      : SAVE #2
6335      034052  006303      :      ASL      R3            : 4
6336      034054  006303      :      ASL      R3            : 8
6337      034056  062603      :      ADD      (SP)+,R3      : (#8)+(#2)=#10.
6338      034060  060203      :      ADD      R2,R3        : UPDATE THE INPUT NUMBER
6339      034062  004037  033736  :      JSR      RD,@CK.CHR      : CHECK ONE CHARACTER
6340      034066  034162      :      BS      : ILLEGAL CHARACTER
6341      034070  034102      :      BS      : CARRIAGE RETURN
6342      034072  034162      :      BS      :
6343      034074  034110      :      BS      :
6344      034076  034106      :      BS      :
6345      034100  034046      :      IS      : DIGIT 0-9
6346      034102  005301      95:      DEC      R1            : BACKUP THE CHARACTER POINTER
6347      034104  000401      :      BR      35          : CONTINUE
6348      034106  005724      25:      TST      (R4)+        : "PERIOD"
6349      034110  005724      35:      TST      (R4)+        : "COMMA" OR "CR"
6350      034112  004037  033736  :      JSR      RD,@CK.CHR      : CHECK ONE CHARACTER
6351      034116  034162      :      BS      : ILLEGAL CHARACTER
6352      034120  034162      :      BS      : CARRIAGE RETURN
6353      034122  034162      :      BS      :
6354      034124  034162      :      BS      :
6355      034126  034132      :      BS      :
6356      034130  034142      :      BS      :
6357      034132  005724      45:      TST      (R4)+        : DIGIT 0-9
6358      034134  005711      :      TSTB     (R1)        : "PERIOD-PERIOD"
6359      034136  001405      :      BEQ     65          : "CR"?
6360      034140  007410      :      BR      65          : YES--BRANCH
6361      034142  126127  177776 000054 55:      CMPB     -2(R1),#',      : WAS CHARACTER BEFORE THE DIGIT A COMMA?
6362      034150  001004      :      BNE     65          : NO--EXIT
6363      034152  020316      65:      CMP      R3,(SP)      : INPUT TO LARGE?
6364      034154  101001      :      BHI     75          : YES -- BRANCH
6365      034156  060400      :      ADD     R4,RD        : ADJUST RETURN ADDRESS
6366      034160  005720      75:      TST      (R0)+        :
6367      034162  010302      85:      MOV      R3,R2        : NUMBER TO R2
6368      034164  005726      :      TST     (SP)+        : CLEAN MAX. SIZE OFF OF STACK
6369      034166  012603      :      MOV     (SP)+,R3     : RESTORE R3
6370      034170  012604      :      MOV     (SP)+,R4     : RESTORE R4

```



```

6371 034172 011000      MOV      (R0),R0      ;GET RETURN ADDRESS
6372 034174 000200      RTS        R0        ;RETURN
6373
6374                ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
6375                ;AND FORMS AN OCTAL NUMBER IN R2
6376                ;CALL:
6377                ;
6378                ;
6379                ;
6380                ;
6381                ;
6382                ;
6383                ;
6384 034176 010346      CK.NUM: MOV      R3,-(SP)      ;SAVE R3
6385 034200 005003      CLR      R3          ;START NUMBER AT ZERO
6386 034202 004037 033662  JSR      R0,@CK.NUM  ;GO FORM THE NUMBER
6387 034206 000440      BR      6$          ;OCTAL DIGIT?
6388 034210 005201      1$: INC     R1        ;NO--BRANCH
6389 034212 006303      ASL     R3          ;MOVE TO NEXT CHARACTER
6390 034214 103435      BCS    6$          ;FOR THE OCTAL NUMBER IN R3
6391 034216 006303      ASL     R3          ;DON'T LET IT GET TO BIG
6392 034220 103433      BCS    6$
6393 034222 006303      ASL     R3
6394 034224 103431      BCS    6$
6395 034226 060203      ADD     R2,R3
6396 034230 004037 033662  JSR      R0,@CK.OCT  ;IS THIS AN OCTAL DIGIT?
6397 034234 000401      BR      2$          ;NO--FIND OUT WHAT IT IS
6398 034236 000764      BR      1$          ;YES--MAKE IT PART OF THE NUMBER
6399 034240 010302      2$: MOV     R3,R2    ;SAVE THE OCTAL NUMBER
6400 034242 005003      CLR     R3          ;START WITH ZERO INDEX
6401 034244 004037 033736  JSR      R0,@CK.CHR  ;CHECK ONE CHARACTER
6402 034250 034310      6$     ;ILLEGAL CHARACTER
6403 034252 034300      5$     ;CARRIAGE RETURN
6404 034254 034310      6$     ;
6405 034256 034300      5$     ;
6406 034260 034264      3$     ;
6407 034262 034310      6$     ;
6408 034264 005723      3$: TST    (R3)+     ;DIGIT 0-9
6409 034266 121127 000056  CMPB   (R1),R'.'    ;"PERIOD"
6410 034272 001002      BNE    5$          ;"PERIOD-PERIOD"?
6411 034274 005201      INC     R1        ;NO--BRANCH
6412 034276 005723      4$: TST    (R3)+     ;YES--ADVANCE THE POINTER
6413 034300 005723      5$: TST    (R3)+     ;"PERIOD-PERIOD"
6414 034302 105711      TSTB  (R1)        ;"COMMA"
6415 034304 001001      BNE    6$          ;"CR"?
6416 034306 060300      ADD     R3,R0     ;NO--BRANCH
6417 034310 012603      6$: MOV     (SP)+,R3  ;YES--SAVE THE OCTAL NUMBER
6418 034312 011000      MOV     (R0),R0   ;RESTORE R3
6419 034314 000200      RTS     R0        ;PICKUP EXIT ADDRESS
                    ;RETURN
    
```


6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475

;;*****

.SBTTL SINGLE/DUAL PORT RH11/RPO4/5/6 DRIVER (REV 1.0)

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

;;*****

;STORAGE FOR RPDS1, RPER1, RPER2, AND RPER3 ON AN ERROR "2"

;RPERRS = RPDS1
;RPERRS+2 = RPER1
;RPERRS+4 = RPER2
;RPERRS+6 = RPER3

034316 000000 000000 000000 RPERRS: .WORD 0,0,0,0
034324 000000

;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)

;DRVACT=0 IF DRIVE IS IDLE
;DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
;DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION

034326 000
034327 000
034330 000
034331 000
034332 000
034333 000
034334 000
034335 000
DRVACT: .BYTE 0 ;DRIVE 0
.BYTE 0 ;DRIVE 1
.BYTE 0 ;DRIVE 2
.BYTE 0 ;DRIVE 3
.BYTE 0 ;DRIVE 4
.BYTE 0 ;DRIVE 5
.BYTE 0 ;DRIVE 6
.BYTE 0 ;DRIVE 7

;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)

;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXISTENT
;DRVSTA>0 IF DRIVE IS ONLINE
;DRVSTA<0 IF DRIVE IS UNSAFE

034336 000
034337 000
034340 000
034341 000
034342 000
034343 000
034344 000
034345 000
DRVSTA: .BYTE 0 ;DRIVE 0
.BYTE 0 ;DRIVE 1
.BYTE 0 ;DRIVE 2
.BYTE 0 ;DRIVE 3
.BYTE 0 ;DRIVE 4
.BYTE 0 ;DRIVE 5
.BYTE 0 ;DRIVE 6
.BYTE 0 ;DRIVE 7

;TABLE OF DRIVE TYPES (DRVTYPE=8 BYTES)

;DRVTYPE=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
;DRVTYPE=1 IF DRIVE IS RPO4
;DRVTYPE=2 IF DRIVE IS RPOS
;DRVTYPE=4 IF DRIVE IS RPO6
;DRVTYPE=-1 IF NOT RPO4/5/6

G10

```

6476 034346 000
6477 034347 000
6478 034350 000
6479 034351 000
6480 034352 000
6481 034353 000
6482 034354 000
6483 034355 000
6484
6485
6486
6487
6488
6489 034356 000
6490 034357 000
6491 034360 000
6492 034361 000
6493 034362 000
6494 034363 000
6495 034364 000
6496 034365 000
6497
6498
6499
6500
6501
6502 034366 000
6503 034367 000
6504 034370 000
6505 034371 000
6506 034372 000
6507 034373 000
6508 034374 000
6509 034375 000
6510
6511
6512
6513
6514
6515 034376 000000
6516
6517
6518
6519
6520
6521
6522
6523 034400 000000
6524
6525
6526
6527
6528
6529 034402 000
6530
6531
  
```

```

DRV TYP: .BYTE 0           ;DRIVE 0
          .BYTE 00          ;DRIVE 1
          .BYTE 00          ;DRIVE 2
          .BYTE 00          ;DRIVE 3
          .BYTE 00          ;DRIVE 4
          .BYTE 00          ;DRIVE 5
          .BYTE 00          ;DRIVE 6
          .BYTE 0           ;DRIVE 7
  
```

```

;TABLE OF DUAL PORT INITIALIZATION INDICATORS
;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
;DPINT<0 IF INITIALIZATION IS IN PROGRESS
  
```

```

DPINT: .BYTE 0           ;DRIVE 0
        .BYTE 00          ;DRIVE 1
        .BYTE 00          ;DRIVE 2
        .BYTE 00          ;DRIVE 3
        .BYTE 00          ;DRIVE 4
        .BYTE 00          ;DRIVE 5
        .BYTE 00          ;DRIVE 6
        .BYTE 0           ;DRIVE 7
  
```

```

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

```

DPRQS: .BYTE 0           ;DRIVE 0
        .BYTE 00          ;DRIVE 1
        .BYTE 00          ;DRIVE 2
        .BYTE 00          ;DRIVE 3
        .BYTE 00          ;DRIVE 4
        .BYTE 00          ;DRIVE 5
        .BYTE 00          ;DRIVE 6
        .BYTE 0           ;DRIVE 7
  
```

```

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

```

TRNSWT: .WORD 0
  
```

```

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

```

SRCHWT: .WORD 0
  
```

```

;RPO4/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
;ACTDRV=0 IF DRIVER IS INACTIVE
;ACTDRV>0 IF DRIVER IS ACTIVE
  
```

```

ACTDRV: .BYTE 0
  
```

```

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


H10

MD-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 125
DZRJA.014 19-APR-76 00:00 SINGLE/DUAL PORT RH11/RPO4/5/6 DRIVER (REV 1.0)

```
6532                                     ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
6533                                     ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
6534
6535 034403      000      ACTSTR: .BYTE  0
6536
6537                                     ;UNLOAD FLAG (ULDFLG=8 BYTES)
6538                                     ;ULDFLG=0 IF NO UNLOAD COMMAND
6539                                     ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
6540                                     ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE
6541
6542 034404      000      ULDFLG: .BYTE  0          ;DRIVE 0
6543 034405      000      .BYTE  0          ;DRIVE 1
6544 034406      000      .BYTE  0          ;DRIVE 2
6545 034407      000      .BYTE  0          ;DRIVE 3
6546 034410      000      .BYTE  0          ;DRIVE 4
6547 034411      000      .BYTE  0          ;DRIVE 5
6548 034412      000      .BYTE  0          ;DRIVE 6
6549 034413      000      .BYTE  0          ;DRIVE 7
6550
6551                                     ;LOOK AHEAD COUNT (LACNT=8 BYTES)
6552                                     ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
6553
6554 034414      000      LACNT:  .BYTE  0          ;DRIVE 0
6555 034415      000      .BYTE  0          ;DRIVE 1
6556 034416      000      .BYTE  0          ;DRIVE 2
6557 034417      000      .BYTE  0          ;DRIVE 3
6558 034420      000      .BYTE  0          ;DRIVE 4
6559 034421      000      .BYTE  0          ;DRIVE 5
6560 034422      000      .BYTE  0          ;DRIVE 6
6561 034423      000      .BYTE  0          ;DRIVE 7
6562
6563                                     ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
6564                                     ;SAVEFG <0 IF SAVE THE RH11/RPO4/5/6 REGISTERS WHEN THE
6565                                     ;OPERATION IS COMPLETED AS PER (DPB+14).
6566                                     ;SAVEFG=0 IF SAVE THE RH11/RPO4/5/6 REGISTERS, AS PER
6567                                     ;(DPB+14), AFTER AN ERROR.
6568
6569 034424      000000   SAVEFG: .WORD  0
6570
6571                                     ;SEEK FLAG (SEEKFG=1 WORD)
6572                                     ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
6573                                     ;FOR A DATA TRANSFER START A SEARCH COMMAND
6574                                     ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
6575                                     ;DISREGARD THE WINDOW
6576
6577 034426      000000   SEEKFG: .WORD  0
6578
6579                                     ;TIMEOUT TABLE (TIMER=8 WORDS)
6580                                     ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
6581
6582 034430      177777   TIMER:  .WORD -1          ;DRIVE 0
6583 034432      177777   .WORD -1          ;DRIVE 1
6584 034434      177777   .WORD -1          ;DRIVE 2
6585 034436      177777   .WORD -1          ;DRIVE 3
6586 034440      177777   .WORD -1          ;DRIVE 4
6587 034442      177777   .WORD -1          ;DRIVE 5
```


6588 034444 177777
 6589 034446 177777
 6590
 6591
 6592
 6593
 6594
 6595 034450 177777
 6596
 6597
 6598
 6599
 6600
 6601 034452 001
 6602 034453 002
 6603 034454 004
 6604 034455 010
 6605 034456 020
 6606 034457 040
 6607 034460 100
 6608 034461 200
 6609
 6610
 6611
 6612
 6613 034462 000003
 6614
 6615
 6616
 6617
 6618 034464 176700
 6619 034466 000254 000240
 6620
 6621
 6622
 6623 034472 000004
 6624
 6625
 6626 034474 001000
 6627
 6628
 6629 034476 000200
 6630
 6631
 6632 034500 000005
 6633
 6634
 6635
 6636 000000
 6637 000002
 6638 000004
 6639 000006
 6640 000010
 6641 000012
 6642 000014
 6643 000016

```

        .WORD -1      ;DRIVE 6
        .WORD -1      ;DRIVE 7

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

DTUM: .WORD -1

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

ATABIT: .BYTE 1      ;DRIVE 0
        .BYTE 2      ;DRIVE 1
        .BYTE 4      ;DRIVE 2
        .BYTE 10     ;DRIVE 3
        .BYTE 20     ;DRIVE 4
        .BYTE 40     ;DRIVE 5
        .BYTE 100    ;DRIVE 6
        .BYTE 200    ;DRIVE 7

;RPO4/5/6 TO RH11 "MSSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
;CALLING IT FATAL (MCPENX=1 WORD)

MCPENX: .WORD 3

;STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RPO4/5/6),
;RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).

RPADR: .WORD 176700
RPVEC: .WORD 254,5*32.

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

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

MXDLTA: .WORD 8*64.
;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

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

MXWINDW: .WORD 5

;DEFINITIONS OF THE RH11/RPO4/5/6 ADDRESS INDEXES

RPCS1=0      ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
RPMC=2      ;WORD COUNT REGISTER (NOT A DRIVE REG)
RPBA=4      ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
RPDA=6      ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
RPCS2=10    ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
RPDS1=12    ;DRIVE STATUS REGISTER (DRIVE REG 01)
RPER1=14    ;ERROR REGISTER #1 (DRIVE REG. 02)
RPAS=16     ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
    
```


6644	000020	RPLA=20	:LOOK AHEAD REGISTER (DRIVE REG. 07)
6645	000022	RPOB=22	:DATA BUFFER REGISTER (NOT A DRIVE REG.)
6646	000024	RPFR=24	:MAINTAINABILITY REGISTER (DRIVE REG. 03)
6647	000026	RPDT=26	:DRIVE TYPE REGISTER (DRIVE REG. 06)
6648	000030	RPSN=30	:SERIAL NUMBER REGISTER (DRIVE REG. 10)
6649	000032	RPOF=32	:OFFSET REGISTER (DRIVE REG. 11)
6650	000034	RPCA=34	:DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
6651	000036	RPCC=36	:CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
6652	000040	RPER2=40	:ERROR REGISTER #2 (DRIVE REG. 14)
6653	000042	RPER3=42	:ERROR REGISTER #3 (DRIVE REG. 15)
6654	000044	RPEC1=44	:ECC POSITION REGISTER (DRIVE REG. 16)
6655	000046	RPEC2=46	:ECC PATTERN REGISTER (DRIVE REG. 17)

;RH11/RPO4/5/6 DRIVER INITIALIZATION CODE
 ;THIS ROUTINE WILL DETERMINE WHICH RPO4/5/6 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

NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED

6670	034502	104412	RPINIT: SAVREG	:SAVE R0 - R5
6671	034504	013746	MOV #PS, -(SP)	:SAVE THE PRESENT PROCESSOR STATUS
6672	034510	012737	MOV #(<5*32.>), #PS	:CHANGE THE PRIORITY TO 5
6673	034516	004737	JSR PC,CLIQUE	:CLEAR ALL REQUEST QUEUES
6674	034522	012701	MOV #RPER5,R1	:FIRST ADDRESS TO BE CLEARED
6675	034526	012702	MOV #SEEKFG,R2	:LAST ADDRESS TO BE CLEARED
6676	034532	005021	1S: CLR (R1)+	:CLEAR
6677	034534	020102	CMR R1,R2	:ARE WE DONE?
6678	034536	101775	BLOS 1S	:BRANCH IF NO
6679	034540	012702	MOV #DTUM,R2	:LAST ADDRESS
6680	034544	012721	2S: MOV #-1,(R1)+	:INITIALIZE
6681	034550	020102	CMR R1,R2	:DONE?
6682	034552	101774	BLOS 2S	:LOOP IF NO
6683	034554	005037	CLR DRVSTA	:SET ALL DRIVES TO OFFLINE
6684	034560	005037	CLR DRVSTA+2	
6685	034564	005037	CLR DRVSTA+4	
6686	034570	005037	CLR DRVSTA+6	
6687	034574	013703	MOV RPVEC,R3	:SETUP THE RH11/RPO4/5/6 VECTOR
6688	034600	012723	MOV #ISR,(R3)+	
6689	034604	013713	MOV RPVEC+2,(R3)	
6690	034610	013704	MOV RPADR,R4	:FIRST ADDRESS OF RH11/RPO4
6691	034614	012764	MOV #BIT05,RPCS2(R4)	:MASSBUS INIT
6692	034622	005001	CLR R1	:START WITH DRIVE 0
6693	034624	004037	3S: JSR R0,DRVINT	:INIT THE DRIVE
6694	034630	000401	BR 4S	: 'DVA' NOT SET OR PARITY ERROR
6695	034632	000402	BR 5S	:NORMAL RETURN
6696	034634	105061	4S: CLRB DRVSTA(R1)	:SET DRIVE STATUS TO OFFLINE
6697	034640	005201	5S: INC R1	:GO TO NEXT DRIVE
6698	034642	042701	BIC #1C7,R1	:MASK OUT UNUSED BITS
6699	034646	001366	BNE 3S	:BR IF MORE DRIVES TO GO


```

6756 035066 001407          BEQ      2$          ;BR IF YES
6757 035070 022705 024022      CMP      #24022,R5   ;DUAL PORT RPO6 ?
6758 035074 001404          BEQ      2$          ;SM IF YES
6759 035076 112761 177777 034346  MOVB    #-1,DRVTP(R1) ;SET INDICATOR TO 'OTHER'
6760 035104 000446          BR       6$          ;EXIT
6761 035106 012746 000121      2$:     MOV     #121,-(SP)   ;DO A "READ-IN PRESET"
6762 035112 004037 041612      JSR     RO,WRT.RP
6763 035116 000000          RPCSI
6764 035120 035246          BS
6765 035122 012746 010000      MOV     #BIT12,-(SP) ;SET FMT22=1
6766 035126 004037 041612      JSR     RO,WRT.RP
6767 035132 000032          RPOF
6768 035134 035246          BS
6769 035136 004037 041432      JSR     RO,RO.RP    ;READ RPDS1
6770 035142 000012          RPDS1
6771 035144 035246          BS
6772 035146 012605          MOV     (SP)+,R5    ;AND SAVE IT IN R5
6773 035150 100015          BPL     4$          ;BRANCH IF ATA=0
6774 035152 116164 034452 000016  MOVB    ATABIT(R1),RPAS(R4) ;CLEAR ATTENTION BIT
6775 035160 004037 041432      JSR     RO,RO.RP    ;FIND OUT WHY ATA=1
6776 035164 000014          RPER1
6777 035166 035246          BS
6778 035170 006126          ROL     (SP)+      ;IS IT UNSAFE?
6779 035172 100004          BPL     4$          ;BR IF NOT
6780 035174 112761 177777 034336  MOVB    #-1,DRVSTA(R1) ;SET UNSAFE INDICATOR
6781 035202 000407          BR       6$          ;EXIT
6782 035204 005105          COM     R5         ;CHECK MOL, DPR, DRY, AND VV
6783 035206 042705 167077      BIC     #1<BIT12:BIT08:BIT07:BIT06>,R5
6784 035212 001003          BNE     6$          ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
6785 035214 112761 000001 034336  MOVB    #1,DRVSTA(R1) ;SET DRIVE STATUS TO ONLINE
6786 035222 005720          TST    (RO)+      ;STEP OVER THE ERROR RETURN
6787 035224 000410          BR       8$          ;EXIT
6788 035226 006301          ASL    R1         ;CHANGE INDEX TO ADDRESS WORDS
6789 035230 012761 003720 034430  MOV     #2000.,TIMER(R1) ;START 2 SEC TIMER
6790 035236 006201          ASR    R1         ;RESTORE R1
6791 035240 112761 177777 034356  MOVB    #-1,DPINT(R1) ;SET PORT INITIALIZE INIDICATOR
6792 035246 012605          8$:     MOV     (SP)+,R5   ;RESTORE R5
6793 035250 000200          RTS     RO         ;EXIT
6794
6795          ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
6796
6797          ;CALL
6798
6799          JSR     RO,#RPO4 ;CALL THE RPO4/5/6 DRIVER
6800          PNTADR ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
6801          RETURN1 ;RETURN HERE IF QUEUE IS FULL
6802          RETURN2 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
6803          ;IS AN ERROR CONDITION
6804
6805 035252 013746 177776  RPO4:  MOV     #RPS,-(SP) ;SAVE THE CALLING STATUS
6806 035256 013737 034470 177776  MOV     RPVEC+2,#RPS ;DON'T ALLOW ANY RPO4/5/6 INTERRUPTS
6807 035264 112737 000001 034402  MOVB    #1,ACTDRV  ;SET "ACTIVE DRIVER" FLAG
6808 035272 104412          SAVREG ;SAVE RO - R5
6809 035274 011002          MOV     (RO),R2    ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
6810 035276 005062 000016  CLR     16(R2)     ;CLEAR THE STATUS/ERROR INDICATOR
6811 035302 111201          MOVB    (R2),R1    ;PICKUP THE DRIVE NUMBER
    
```


N10

6868	035544	104412			OPT:	SAVREG	;SAVE R0 - R5
6869	035546	013746	177776			MOV	;SAVE PROC. STATUS
6870	035552	146137	034452	034400		BICB	;CLEAR "SEARCH WAIT" KEY
6871	035560	004737	042640			JSR	;GET "DPB" POINTER OF REQUEST
6872	035564	005702				TST	;IS THERE A REQUEST IN QUEUE?
6873	035566	001505				BEQ	;NO--BRANCH TO EXIT
6874	035570	032764	010000	000012		BIT	;IS MOL STILL SET ?
6875	035576	001404				BEQ	;BR IF NOT
6876	035600	032764	000100	000012		BIT	;IS VV SET ?
6877	035606	001003				BNE	;BR IF IT IS
6878	035610	004037	034714		9\$:	JSR	;SEE IF DRIVE STILL ONLINE ?
6879	035614	000470				BR	;PARITY OR 'DVA' NOT SET
6880	035616	105761	034336		10\$:	TSTB	;IS DRIVE ONLINE?
6881	035622	003014				BGT	;YES--BRANCH
6882	035624	004737	042662			JSR	;NO--REMOVE REQUEST FROM QUEUE
6883	035630	012762	140000	000016		MOV	;SET OFFLINE STATUS/ERROR INDICATOR
6884	035636	105761	034336			TSTB	;IS DRIVE UNSAFE ?
6885	035642	100064				BPL	;BR TO EXIT IF NOT
6886	035644	012762	110000	000016		MOV	;SET UNSAFE STATUS/ERROR INDICATOR
6887	035652	000460				BR	;BRANCH TO EXIT
6888	035654	012746	000111		1\$:	MOV	;LOAD COMMAND ONTO THE STACK
6889	035660	004037	041612			JSR	;LOAD THE REGISTER
6890	035664	000000				RPCS1	;REGISTER INCREMENT
6891	035666	035776				6\$;ERROR RETURN ADDRESS
6892	035670	032714	004000			BIT	;DRIVE AVAILABLE ?
6893	035674	001427				BEQ	;BR IF NOT
6894	035676	122762	000150	000002		CMPB	;IS THE REQUEST FOR I/O?
6895	035704	002403				BLT	;YES--BRANCH
6896	035706	004737	036240			JSR	;CALL THE COMMAND INITIATOR
6897	035712	000440				BR	;BRANCH TO EXIT
6898	035714	005737	034450		2\$:	TST	;DATA TRANSFER UNDERWAY?
6899	035720	002012				BGE	;YES--GO START A SEARCH
6900	035722	005737	034426			TST	;DO IMPLIED SEEKS?
6901	035726	100404				BMI	;YES---BRANCH
6902	035730	004037	037210			JSR	;NO--DO LOOK AHEAD
6903	035734	000427				BR	;RETURN HERE ON A PARITY ERROR
6904	035736	000403				BR	;GO START A SEARCH
6905	035740	004737	036024		3\$:	JSR	;START A DATA TRANSFER
6906	035744	000423				BR	
6907	035746	004737	036132		4\$:	JSR	;START A SEARCH
6908	035752	000420				BR	;GO TO THE EXIT
6909	035754	112761	177777	034366	5\$:	MOV	;SET PORT REQUEST INDICATOR
6910	035762	010103				MOV	;SET UP TO ADDRESS WORDS
6911	035764	006303				ASL	;CONVERT TO WORD INDEX
6912	035766	012763	023420	034430		MOV	;START 10 SEC TIMER
6913	035774	000402				BR	;EXIT
6914	035776	004737	036654		6\$:	JSR	;PROCESS THE PARITY ERROR
6915	036002	032714	000100		7\$:	BIT	;SEE IF 'IE' ALREADY SET
6916	036006	001002				BNE	;BR IF SET
6917	036010	004737	042122			JSR	;SET "IE" WITHOUT A "TRE"
6918	036014	012637	177776		8\$:	MOV	;RESTORE PROC. STATUS
6919	036020	104413				RESREG	;RESTORE R0 - R5
6920	036022	000207				RTS	PC
6921							
6922							
6923							

;COMMAND INITIATOR

D11

```

6924          :CALL
6925          :
6926          :
6927          :
6928          :
6929          :
6930          :
6931          :
6932          :
6933 036024 004737 042662      C11: JSR    PC,POPQUE      ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
6934 036030 010237 034376      MOV    R2,TRANSMT    ;PUT REQ. IN TRANSFER WAIT QUEUE
6935 036034 010203              MOV    R2,R3         ;DPB ADDRESS TO R3
6936 036036 013704 034464      MOV    RPADR,R4      ;RPCS1 ADDRESS
6937 036042 010164 000010      MOV    R1,RPCS2(R4) ;SELECT DRIVE
6938 036046 062703 000004      ADD    #4,R3         ;DESIRED WORD COUNT
6939 036052 062704 000002      ADD    #2,R4         ;RPMC ADDRESS
6940 036056 012324              MOV    (R3)+,(R4)+   ;LOAD WORD COUNT
6941 036060 012324              MOV    (R3)+,(R4)+   ;LOAD BUFFER ADDRESS
6942 036062 012346              MOV    (R3)+,-(SP)   ;LOAD SECTOR AND TRACK
6943 036064 004037 041612      JSR    RD,WRT.RP     ;CALL THE LOAD(WRITE) ROUTINE
6944 036070 000006      RPDAR              ;INDEX OF REGISTER TO LOAD
6945 036072 036654      CI7              ;ERROR RETURN ADDRESS
6946 036074 012346              MOV    (R3)+,-(SP)   ;LOAD CYLINDER ADDRESS
6947 036076 004037 041612      JSR    RD,WRT.RP
6948 036102 000034      RPCA              ;
6949 036104 036654      CI7              ;
6950 036106 016246 000002      MOV    2(R2),-(SP)  ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
6951 036112 004037 041612      JSR    RD,WRT.RP
6952 036116 000000      RPCS1            ;
6953 036120 036654      CI7              ;
6954 036122 010137 034450      MOV    R1,DTUM      ;SET "DATA TRANSFER UNDERWAY"
6955 036126 000137 036616      JMP    CIS          ;
6956 036132 013704 034464      C13: MOV    RPADR,R4     ;RPCS1 ADDRESS
6957 036136 010164 000010      MOV    R1,RPCS2(R4) ;SELECT DRIVE
6958 036142 016246 000012      MOV    12(R2),-(SP) ;DESIRED CYLINDER ADDRESS
6959 036146 004037 041612      JSR    RD,WRT.RP
6960 036152 000034      RPCA              ;
6961 036154 036654      CI7              ;
6962 036156 116203 000010      MOV    10(R2),R3    ;PICKUP SECTOR ADDRESS
6963 036162 163703 034500      SUB    MYWINDOW,R3  ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
6964 036166 002002      BGE    15          ;
6965 036170 062703 000026      ADD    #22,R3       ;
6966 036174 010346              MOV    R3,-(SP)     ;COMBINE THE ADJUSTED SECTOR WITH
6967 036176 116266 000011 000001 15: MOV    11(R2),1(SP) ;THE DESIRED TRACK
6968 036204 004037 041612      JSR    RD,WRT.RP    ;LOAD DESIRED TRACK & SECTOR
6969 036210 000006      RPDAR              ;
6970 036212 036654      CI7              ;
6971 036214 012746 000131      MOV    #131,-(SP)   ;START A SEARCH
6972 036220 004037 041612      JSR    RD,WRT.RP
6973 036224 000000      RPCS1            ;
6974 036226 036654      CI7              ;
6975 036230 156137 034452 034400 15: BISB  ATABIT(R1),SRCHWT ;SET "SEARCH WAIT" KEY
6976 036236 000567      BR    CIS          ;
6977 036240 013704 034464      C14: MOV    RPADR,R4     ;RPCS1 ADDRESS
6978 036244 010164 000010      MOV    R1,RPCS2(R4) ;SELECT DRIVE
6979 036250 116203 000002      MOV    2(R2),R3     ;PICKUP THE REQUESTED COMMAND
    
```


6980	036254	122703	000131		CMPB	#131,R3	: IS IT A SEARCH COMMAND?
6981	036260	001007			BNE	1\$: BRANCH IF NO
6982	036262	016246	000010		MOV	10(R2),-(SP)	: LOAD DESIRED TRACK & SECTOR
6983	036266	004037	041612		JSR	RD,WRT.RP	
6984	036272	000006			RPOA		
6985	036274	036654			CI7		
6986	036276	000403			BR	2\$: GO LOAD CYLINDER
6987	036300	122703	000105	1\$:	CMPB	#105,R3	: IS IT A SEEK COMMAND
6988	036304	001007			BNE	3\$: BRANCH IF NO
6989	036306	016246	000012	2\$:	MOV	12(R2),-(SP)	: LOAD DESIRED CYLINDER
6990	036312	004037	041612		JSR	RD,WRT.RP	
6991	036316	000034			RPCA		
6992	036320	036654			CI7		
6993	036322	000546			BR	CI6	
6994	036324	122703	000115	3\$:	CMPB	#115,R3	: IS IT AN "OFFSET" COMMAND?
6995	036330	001013			BNE	4\$: BR IF NO
6996	036332	004037	041432		JSR	RD,RD.RP	: MERGE THE OFFSET VALUE INTO RPOF
6997	036336	000032			RPOF		: BUT DON'T CHANGE THE UPPER
6998	036340	036654			CI7		
6999	036342	116216	000001		MOVW	1(R2),(SP)	: BYTE WHEN LOADING THE
7000	036346	004037	041612		JSR	RD,WRT.RP	: REGISTER (RPOF)
7001	036352	000032			RPOF		
7002	036354	036654			CI7		
7003	036356	000530			BR	CI6	: GO START THE COMMAND
7004	036360	122703	000107	4\$:	CMPB	#107,R3	: IS IT A "RECALIBRATE" COMMAND?
7005	036364	001525			BEQ	CI6	: BRANCH IF YES
7006	036366	122703	000117		CMPB	#117,R3	: IS IT A RETURN TO CENTER?
7007	036372	001522			BEQ	CI6	: BRANCH IF YES
7008	036374	122703	000103		CMPB	#103,R3	: IS IT AN "UNLOAD" COMMAND?
7009	036400	001016			BNE	5\$: BRANCH IF NO
7010	036402	112761	000001	034326	MOVW	#1,DRVACT(R1)	: SET THE DRIVE ACTIVE INDICATOR
7011	036410	105061	034336		CLRB	DRVSTA(R1)	: PUT DRIVE STATUS TO OFFLINE
7012	036414	112761	000001	034404	MOVW	#1,ULDFLG(R1)	: SET "UNLOAD IN PROGRESS" FLAG
7013	036422	010346			MOV	R3,-(SP)	: START THE "UNLOAD" COMMAND
7014	036424	004037	041612		JSR	RD,WRT.RP	
7015	036430	000000			RPCS1		
7016	036432	036654			CI7		
7017	036434	000207			RTS	PC	: RETURN TO USER
7018	036436	122703	000143	5\$:	CMPB	#143,R3	: IS IT A "SET FORMAT" COMMAND?
7019	036442	001014			BNE	6\$: BRANCH IF NO
7020	036444	004037	041432		JSR	RD,RD.RP	: READ THE OFFSET REGISTER
7021	036450	000032			RPOF		
7022	036452	036654			CI7		
7023	036454	116266	000001	000001	MOVW	1(R2),1(SP)	: COMBINE "FMT22", "ECI", AND "HCI"
7024	036462	004037	041612		JSR	RD,WRT.RP	: LOAD "FMT22", "ECI", AND/OR "HCI".
7025	036466	000032			RPOF		
7026	036470	036654			CI7		
7027	036472	000436			BR	12\$	
7028	036474	122703	000141	6\$:	CMPB	#141,R3	: IS IT A "GET REGISTER" COMMAND?
7029	036500	001023			BNE	10\$: BRANCH IF NO
7030	036502	016203	000006	7\$:	MOV	6(R2),R3	: POINTS TO 1ST ADDRESS OF WHERE
7031							: TO PUT THE REGISTER(S)
7032	036506	116237	000010	036524	MOVW	10(R2),9\$: INIT. THE INDEX FOR THE FIRST REG.
7033	036514	116205	000011		MOVW	11(R2),R5	: INDEX OF LAST REG. TO MOVE
7034	036520	004037	041432	8\$:	JSR	RD,RD.RP	: READ RPO4/5/6 REGISTER
7035	036524	000000		9\$:	RPCS1		: INDEX OF REG. TO READ

7036	036526	036654				CI7		
7037	036530	012623				MOV	(SP)+,(R3)+	:GET THE CONTENTS OF RH11/RPO4/5/6 REG.
7038	036532	023705	036524			CMP	96,R5	:LAST REG. BEEN READ?
7039	036536	001414				BEQ	12\$:GET OUT IF YES
7040	036540	062737	000002	036524		ADD	82,9\$:INCREASE THE INDEX BY 2
7041	036546	000764				BR	8\$:LOOP--MORE TO READ
7042	036550	122703	000145		10\$:	CMPB	8145,R3	:IS IT A "SELECT DRIVE" COMMAND?
7043	036554	001405				BEQ	12\$:BRANCH IF YES
7044	036556	010346			11\$:	MOV	R3,-(SP)	:LOAD THE COMMAND
7045	036560	004037	041612			JSR	RD,WRT.RP	
7046	036564	000000				RPCS1		
7047	036566	036654				CI7		
7048	036570	004737	042662		12\$:	JSR	PC,POPQUE	:REMOVE REG. FROM QUEUE
7049	036574	052762	000200	000016		BIS	8BIT07,16(R2)	:SET THE "DONE" BIT
7050	036602	005737	034424			TST	SAVEFG	:SAVE THE RH11/RPO4/5/6 REGISTERS?
7051	036606	100002				BPL	13\$:BRANCH IF NO
7052	036610	004737	042004			JSR	PC,SVRH11	:YES--GO SAVE THE REGISTERS
7053	036614	000207			13\$:	RTS	PC	:RETURN TO USER
7054	036616	006301			CI5:	ASL	R1	
7055	036620	012761	001750	034430		MOV	81000.,TIMER(R1)	:SET A ONE SECOND TIMER
7056	036626	006201				ASR	R1	
7057	036630	112761	000001	034326		MOVB	81,DRVACT(R1)	:SET THE DRIVE ACTIVE
7058	036636	000207				RTS	PC	:RETURN TO THE USER
7059	036640	010346			CI6:	MOV	R3,-(SP)	:LOAD THE COMMAND
7060	036642	004037	041612			JSR	RD,WRT.RP	
7061	036646	000000				RPCS1		
7062	036650	036654				CI7		
7063	036652	000761				BR	CI5	
7064	036654	032764	010000	000010	CI7:	BIT	8BIT12,RPCS2(R4)	:DRIVE NON-EXISTENT ?
7065	036652	001034				BNE	CI8	:BR IF YES
7066	036664	005702			1\$:	TST	R2	:ANYTHING IN QUEUE ?
7067	036666	001405				BEQ	CI7B	:BR IF NOT
7068	036670	012762	104000	000016		MOV	8BIT15!BIT11,16(R2)	:SET "PARITY" ERROR INDICATOR
7069	036676	004737	042004			JSR	PC,SVRH11	:GO SAVE THE RH11/RPO4/5/6 REGISTERS
7070	036702	012746	000111		CI7B:	MOV	8111,-(SP)	:DO A "DRIVE CLEAR"
7071	036706	004037	041612			JSR	RD,WRT.RP	
7072	036712	000000				RPCS1		
7073	036714	036754				CI8		
7074	036716	004737	042544			JSR	PC,EMPTYQ	:EMPTY THE QUEUE
7075	036722	105061	034404			CLRB	ULDFLG(R1)	:CLEAR THE UNLOAD IN QUEUE FLAG
7076	036726	105061	034326			CLRB	DRVACT(R1)	:DRIVE IS IDLE
7077	036732	020137	034450			CMP	R1,DTUM	:IF THIS DRIVE HAD AN I/O REQUEST
7078	036736	001005				BNE	1\$:IN PROGRESS CLEAR ALL OF THE FLAGS
7079	036740	005037	034376			CLR	TRANSMT	
7080	036744	012737	177777	034450		MOV	8-1,DTUM	
7081	036752	000207			1\$:	RTS	PC	
7082	036754	104412			CI8:	SAVREG		:SAVE RD - R5
7083	036756	032764	010000	000010		BIT	8BIT12,RPCS2(R4)	:IS "NED" SET ?
7084	036764	001002				BNE	1\$:BR IF YES
7085	036766	005001				CLR	R1	
7086	036770	005003				CLR	R3	
7087	036772	105761	034326		1\$:	TSTB	DRVACT(R1)	:DRIVE ACTIVE?
7088	036776	001443				BEQ	5\$:BRANCH IF NO
7089	037000	013702	034376			MOV	TRANSMT,R2	:GET THE "TRANSFER WAIT" QUEUE
7090	037004	020137	034450			CMP	R1,DTUM	:DID THIS DRIVE HAVE AN I/O IN PROGRESS?
7091	037010	001402				BEQ	2\$:BRANCH IF YES

F11

MD-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 136
 DZRJA.014 19-APR-76 00:00 SINGLE/DUAL PORT RH11/RPO4/5/6 DRIVER (REV 1.0)

```

7148 037256 000303          SWAB      R3          ;MULT. BY 64--ALIGN WITH
7149 037260 006203          ASR      R3          ;LOOK AHEAD REGISTER
7150 037262 006203          ASR      R3
7151 037264 012737 000340 177776 MOV      #340,2#PS    ;PRIORITY LEVEL "7"
7152 037272 004037 041432 JSR      RD,RD.RP    ;READ LOOK AHEAD REGISTER
7153 037276 000020          RPLA
7154 037300 037340          4S
7155 037302 162603          SUB      (SP)+,R3    ;CALCULATE THE DELTA
7156 037304 002002          BGE      1S
7157 037306 062703 002600 ADD      #(<22.#64.>),R3 ;MAKE THE DELTA POSITIVE
7158 037312 023703 034474 1S:      CMP      MxDLTA,R3 ;CHECK THE DELTA TO SEE
7159 037316 002406          BLT      3S          ;IF IT IS WITHIN THE
7160 037320 023703 034476          CMP      MNDLTA,R3 ;WINDOW---IF YES, ZERO
7161 037324 002003          BGE      3S          ;THE LOOK AHEAD COUNT
7162 037326 105061 034414 2S:      CLRB    LACNT(R1) ;AND TAKE THE I/O EXIT
7163 037332 005720          TST     (R0)+
7164 037334 005720 3S:      TST     (R0)+ ;ADJUST THE RETURN ADDRESS
7165 037336 000402          BR      5S          ;EXIT
7166 037340 004737 036654 4S:      JSR      PC,C17 ;PROCESS THE ERROR
7167 037344 000200 5S:      RTS     RD      ;RETURN
7168
7169 ;INTERRUPT SERVICE ROUTINE
7170
7171 037346 112737 000001 034402 ISR:     MOVB    #1,ACTDRV ;SET "ACTIVE DRIVER" FLAG
7172 037354 104412          SAVREG ;SAVE R0 - R5
7173 037356 013704 034464          MOV     RPADR,R4 ;ADDRESS OF RHCSI
7174 037362 013701 034450          MOV     DTUM,R1 ;GET "DATA TRANSFER UNDERWAY" INDICATOR
7175 037366 002403          BLT     1S          ;BRANCH IF NO DATA TRANSFER UNDERWAY
7176 037370 004737 037412          JSR     PC,TD      ;CALL TRANSFER DONE
7177 037374 000402          BR      2S
7178 037376 004737 037552 1S:      JSR     PC,SC ;CALL SPECIAL CONDITIONS
7179 037402 104413 2S:      RESREG ;RESTORE R0 - R5
7180 037404 105037 034402          CLRB    ACTDRV ;CLEAR "ACTIVE DRIVER" FLAG
7181 037410 000002          RTI     ;RETURN
7182
7183 ;TRANSFER DONE ROUTINE
7184
7185 037412 105061 034326 TD:      CLRB    DRVACT(R1) ;SET DRIVE ACTIVE INDICATOR TO IDLE
7186 037416 012737 177777 034450          MOV     #0-1,DTUM ;NO DATA TRANSFERS UNDERWAY
7187 037424 006301          ASL     R1
7188 037426 012761 177777 034430          MOV     #0-1,TIMER(R1) ;CANCEL TIMEOUT
7189 037434 006201          ASR     R1
7190 037436 013702 034376          MOV     TRNSWT,R2 ;GET "DPB" ADDRESS FROM THE
7191 037442 005037 034376          CLR     TRNSWT ;TRANSFER WAIT QUEUE--CLEAR QUEUE
7192 037446 052762 000200 000016          BIS     @BIT07,16(R2) ;SET DONE
7193 037454 010164 000010          MOV     R1,RPCS2(R4) ;SELECT THE DRIVE
7194 037460 004037 041432          JSR     RD,RD.RP    ;TRANSFER ERROR(TRE=1)?
7195 037464 000000          RPCS1
7196 037466 036654          C17
7197 037470 006126          ROL     (SP)+
7198 037472 100413          BMI     3S
7199 037474 005737 034424          TST     SAVEFG ;BR IF YES
7200 037500 100002          BPL     1S          ;SAVE THE RH11/RPO4/5/6 REGISTERS?
7201 037502 004737 042004          JSR     PC,SVRH11 ;BRANCH IF NO
7202 037506 004737 035544 1S:      JSR     PC,OPT ;YES--SAVE THE REGISTERS
7203 037512 000417          BR      SC          ;CALL OPTIMIZER
                          ;CHECK OTHER DRIVES

```



```

7204 037514 012714 000113      2S:  MOV      #113,(R4)      ;RELEASE THE DRIVE
7205 037520 000414              BR          SC           ;CHECK FOR OTHER DRIVES
7206 037522 052762 100100 000016 3S:  BIS      #BIT15:BIT06,16(R2) ;SET DATA ERROR FLAG
7207 037530 004737 042544              JSR      PC,EMPTYQ      ;EMPTY THE "DRIVE'S WAIT" QUEUE
7208 037534 004737 042004              JSR      PC,SVRH11      ;SAVE THE RH11/RPO4/5/6 REGISTERS
7209 037540 012714 040111              MOV      #40111,(R4)    ;ISSUE A "DRIVE CLEAR"
7210 037544 012714 000113              MOV      #113,(R4)      ;ISSUE A RELEASE TO THE DRIVE
7211 037550 000400              BR          SC           ;CHECK FOR OTHER DRIVES
7212
7213                               ;SPECIAL CONDITION ROUTINE
7214
7215 037552 116403 000016      SC:  MOVB     RPAS(R4),R3    ;READ "RPAS"
7216 037556 001014              BNE     2S              ;BRANCH IF ANY 'ATA' BITS SET
7217 037560 004037 041432              JSR     RD,RD.RP       ;READ CONTROL AND STATUS REGISTER
7218 037564 000000              RPCS1
7219 037566 036754              CIB
7220 037570 106126              ROLB     (SP)+          ;IS "IE"=1?
7221 037572 100405              BMI     1S              ;YES NO DRIVES TO CHECK
7222 037574 004037 042724              JSR     RD,ES.SAV      ;SAVE THE ADDRESS IN 'SESCAPE'
7223 037600 104001              ERROR    1              ;REPORT AN ILLEGAL INTERRUPT
7224 037602 004737 042122              JSR     PC,SET.IE      ;SET INTERRUPT ENABLE
7225 037606 000207              1S:  RTS     PC           ;RETURN
7226 037610 005046              2S:  CLR     -(SP)        ;PROCESS ALL DRIVES THAT HAVE
7227 037612 110316              MOVB     R3,(SP)        ;AN "ATA"=1
7228 037614 012703 000001              MOV     #1,R3
7229 037620 005001              CLR     R1
7230 037622 030316      SC3:  BIT     R3,(SP)        ;ATA=1?
7231 037624 001005              BNE     SC5             ;YES--BRANCH
7232 037626 005201      SC4:  INC     R1           ;MOVE TO THE NEXT DRIVE
7233 037630 106303              ASLB    R3
7234 037632 001373              BNE     SC3            ;BRANCH IF MORE TO CHECK?
7235 037634 005726              TST     (SP)+          ;CLEAN OFF THE STACK
7236 037636 000207              RTS     PC             ;RETURN TO USER
7237 037640 105761 034356      SC5:  TSTB    DPINT(R1)    ;INITIALIZING THE DRIVE ?
7238 037644 001402              BEQ     1S             ;BR IF NOT
7239 037646 000137 040544              JMP     SC13           ;PROCESS THE DRIVE
7240 037652 105761 034366      1S:  TSTB    DPRQS(R1)    ;PORT REQUEST OUTSTANDING ?
7241 037656 001402              BEQ     2S             ;BR IF NOT
7242 037660 000137 040544              JMP     SC13           ;START THE OUTSTANDING COMMAND
7243 037664 105761 034336      2S:  TSTB    DRVSTA(R1)   ;CHECK THE DRIVE STATUS
7244 037670 003025              BGT     5S             ;BRANCH IF ONLINE
7245 037672 105761 034404              TSTB    ULDFLG(R1)    ;UNLOAD IN PROGRESS?
7246 037676 003422              BLE     5S             ;BRANCH IF NOT
7247 037700 004737 042640              JSR     PC,GETREQ      ;GET DPB POINTER
7248 037704 004737 042004              JSR     PC,SVRH11      ;SAVE THE RH11/RPO4/5/6 REGISTERS
7249 037710 004737 040474              JSR     PC,SC12        ;SAVE RPO51, RPER1, RPER2, AND RPER3
7250
7251 037714 105761 034336              TSTB    DRVSTA(R1)    ;ALSO DO A DRIVE INIT (DRVINT)
7252 037720 003416              BLE     6S             ;DID DRIVE COME ONLINE?
7253 037722 032737 040000 034316      BIT     #BIT14,RPER5    ;NO---BRANCH
7254 037730 001002              BNE     3S             ;WAS THERE AN ERROR?
7255 037732 000137 040404              JMP     SC11           ;BR IF ERROR
7256 037736 013705 034320      3S:  MOV     RPER5+2,R5    ;NO ERROR
7257 037742 000502              BR      SC6A           ;YES -- PICKUP RPER1 AND
7258 037744 105761 034326      5S:  TSTB    DRVACT(R1)   ;GO PROCESS THE ERROR
7259 037750 001033              BNE     SC6            ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
                          ;BR IF EITHER

```


7260	037752	004737	040474		JSR	PC,SC12	:SAVE RPO51, RPER1, RPER2, AND RPER3
7261							:ALSO DO A DRVINT
7262	037756	105761	034356	6S:	TSTB	DPINT(R1)	:TRYING TO INIT THE DRIVE ?
7263	037762	001321			BNE	SC4	:BR IF YES, CHECK ON MORE DRIVES
7264	037764	105761	034336		TSTB	DRVSTA(R1)	:CHECK ON DRIVE'S STATUS
7265	037770	100412			BMI	7S	:BR IF UNSAFE
7266	037772	032737	020000	034324	BIT	#BIT13,RPERRS+6	:ADDRESS PLUG CHANGED ?
7267	040000	001013			BNE	8S	:BR IF YES
7268	040002	012746	000113		MOV	#113,-(SP)	:RELEASE COMMAND
7269	040006	004037	041612		JSR	RD,WRT.RP	:WRITE THE COMMAND INTO RPCS!
7270	040012	000000			RPCS1		:REGISTER INDEX
7271	040014	040354			SCB		:PARITY EXIT ADDRESS
7272	040016	011605		7S:	MOV	(SP),RS	:PICKUP (RPAS) BEFORE THE ERROR CALL
7273	040020	004037	042724		JSR	RD,ES.SAV	:SAVE THE ADDRESS IN 'SESCAPE'
7274	040024	104002			ERROR	2	:REPORT THE UNEXPECTED ATTENTION
7275	040026	000677			BR	SC4	:GO CHECK FOR MORE ATA'S
7276	040030			8S:			
7277	040030	004037	042724		JSR	RD,ES.SAV	:SAVE THE ADDRESS IN 'SESCAPE'
7278	040034	104005			ERROR	5	:REPORT THE ADDRESS PLUG CHANGE
7279	040036	000673			BR	SC4	:CHECK FOR MORE DRIVES
7280	040040	006301		SC6:	ASL	R1	:SETUP TO ADDRESS WORDS
7281	040042	012761	177777	034430	MOV	#-1,TIMER(R1)	:STOP THE TIMER
7282	040050	006201			ASR	R1	:RESTORE THE DRIVE ADDRESS
7283	040052	004737	042640		JSR	PC,GETREQ	:GET THE DPB POINTER FROM THE QUEUE
7284	040056	010164	000010		MOV	R1,RPCS2(R4)	:SELECT DRIVE
7285	040062	004037	041432		JSR	RD,RD.RP	:READ THE RPO4'S STATUS REG.
7286	040066	000012			RPO51		
7287	040070	040354			SCB		
7288	040072	011605			MOV	(SP),RS	:AND PUT IT IN RS
7289	040074	006126			ROL	(SP)+	:WAS THERE AN ERROR?
7290	040076	100407			BMI	1S	:BR IF ERROR
7291	040100	105761	034326		TSTB	DRVACT(R1)	:CHECK DRIVE'S STATE
7292	040104	003137			BGT	SC11	:BR IF DRIVE ACTIVE WITH ORDER
7293	040106	052762	100210	000016	BIS	#BIT15!BIT07!BIT03,16(R2)	:INFORM USER OF ERROR RECOVER COMPLETION
7294	040114	000470			BR	SC7	
7295	040116	004037	041432	1S:	JSR	RD,RD.RP	:READ ERROR REGISTER #1
7296	040122	000014			RPER1		
7297	040124	040354			SCB		
7298	040126	012605			MOV	(SP)+,RS	:AND SAVE IT IN RS
7299	040130	004737	042004		JSR	PC,SVRH11	:SAVE RH11/RPO4/5/6 REGISTERS
7300	040134	012746	000111		MOV	#111,-(SP)	:ISSUE A DRIVE CLEAR
7301	040140	004037	041612		JSR	RD,WRT.RP	
7302	040144	000000			RPCS1		
7303	040146	040354			SCB		
7304	040150	006105		SC6A:	ROL	RS	:WAS "UNSAFE" CONDITION =1?
7305	040152	100406			BMI	1S	:BRANCH IF YES
7306	040154	005702			TST	R2	:ANYTHING IN QUEUE ?
7307	040156	001447			BEQ	SC7	:BR IF NOT
7308	040160	052762	100240	000016	BIS	#BIT15!BIT07!BIT05,16(R2)	:INFORM USER OF ERROR
7309	040166	000443			BR	SC7	
7310	040170	004037	041432	1S:	JSR	RD,RD.RP	:READ DRIVE STATUS REG. #1
7311	040174	000012			RPO51		
7312	040176	040354			SCB		
7313	040200	011605			MOV	(SP),RS	:SAVE RPO51 IN RS
7314	040202	006126			ROL	(SP)+	: "ERR"=1?
7315	040204	100011			BPL	2S	:BR IF NO--UNSAFE CLEARED


```

7316 040206 112761 177777 034336      MOVB      0-1,DRVSTA(R1) ;DRIVE IS UNSAFE
7317 040214 004737 042004      JSR       PC,SVRH11    ;SAVE RH11/RPO4/5/6 REGISTERS
7318 040220 052762 110000 000016      BIS      #BIT15!BIT12,16(R2) ;INFORM USER OF UNSAFE ERROR
7319 040226 000423          BR        SC7
7320 040230 032705 010000      2S:     BIT      #BIT12,RS      ;"MOL" = 1 ?
7321 040234 001015          BNE      3S            ;BR IF YES
7322 040236 112761 177777 034326      MOVB      0-1,DRVACT(R1) ;ACTIVE ERROR RECOVER
7323 040244 112761 000001 034336      MOVB      01,DRVSTA(R1) ;ONLINE
7324 040252 006301          ASL      R1
7325 040254 012761 072460 034430      MOV      #30000.,TIMER(R1) ;START 30 SECOND TIMER
7326 040262 006201          ASR      R1
7327 040264 000137 037626      JMP      SC4
7328 040270 052762 100220 000016      3S:     BIS      #BIT15!BIT07!BIT04,16(R2) ;INFORM USER OF ERROR
7329 040276 105061 034326      SC7:     CLRB     DRVACT(R1) ;DRIVE IS IDLE
7330 040302 004737 042544      JSR      PC,EMPTYQ    ;DUMP THE QUEUE
7331 040306 105761 034404      TSTB     ULDFLG(R1)  ;UNLOAD IN PROGRESS OR QUEUE?
7332 040312 003002          BGT      1S            ;BR IF NOT
7333 040314 105061 034404      CLRB     ULDFLG(R1)  ;CLEAR UNLOAD FLAG
7334 040320 116164 034452 000016      1S:     MOVB     ATABIT(R1),RPAS(R4) ;CLEAR ATTENTION BIT
7335 040322 105761 034336      TSTB     DRVSTA(R1)  ;IS THE DRIVE UNSAFE ?
7336 040332 100406          BMI     2S            ;BR IF IT IS
7337 040334 012746 000113      MOV      #113,-(SP)  ;RELEASE COMMAND
7338 040340 004037 041612      JSR      RD,WRT.RP   ;WRITE THE COMMAND INTO RPCS1
7339 040344 000000          RPCS1
7340 040346 040354          SC8
7341 040350 000137 037626      2S:     JMP      SC4          ;CHECK FOR MORE DRIVES
7342 040354 105761 034326      SC8:     TSTB     DRVACT(R1) ;IS DRIVE IDLE?
7343 040360 001405          BEQ     1S            ;YES--BRANCH
7344 040362 004737 042640      JSR      PC,GETREQ   ;GET DPB POINTER
7345 040366 004737 036654      JSR      PC,C17     ;PROCESS THE PARITY ERROR
7346 040372 000402          BR      2S            ;CONTINUE
7347 040374 004737 036702      1S:     JSR      PC,C17B ;PROCESS THE UNCORRECTABLE PARITY ERROR
7348 040400 000137 037626      2S:     JMP      SC4          ;CHECK MORE DRIVES
7349 040404 105761 034404      SC11:    TSTB     ULDFLG(R1) ;"UNLOAD IN PROGRESS"?
7350 040410 003402          BLE     1S            ;BRANCH IF NO
7351 040412 105061 034404      CLRB     ULDFLG(R1)  ;CLEAR UNLOAD FLAG
7352 040416 105061 034326      1S:     CLRB     DRVACT(R1) ;SET DRIVE IDLE
7353 040422 136137 034452 034400      BITB     ATABIT(R1),SRCHMT ;DOING A SEARCH OPERATION FOR
7354          ;AN I/O COMMAND?
7355 040430 001012          BNE     2S            ;BRANCH IF YES
7356 040432 004737 042662          JSR      PC,POPQUE  ;REMOVE REQUEST FROM QUEUE
7357 040436 052762 000200 000016      BIS      #BIT07,16(R2) ;SET "DONE" BIT
7358 040444 005737 034424      TST      SAVEFG     ;SAVE THE REGISTERS?
7359 040450 100002          BPL     2S            ;BRANCH IF NO
7360 040452 004737 042004          JSR      PC,SVRH11  ;YES--SAVE ALL OF THE RH11/RPO4/5/6 REG'S
7361 040456 116164 034452 000016      2S:     MOVB     ATABIT(R1),RPAS(R4) ;CLEAR ATTENTION BIT
7362 040464 004737 035544      JSR      PC,OPT     ;START A REQUEST
7363 040470 000137 037526      JMP      SC4          ;CHECK FOR MORE DRIVES
7364 040474 010164 000010      SC12:    MOV      R1,RPCS2(R4) ;SELECT DRIVE
7365 040500 016437 000012 034316      MOV      RPOS1(R4),RPERRS ;SAVE THE FOUR REGISTERS THAT
7366 040506 016437 000014 034320      MOV      RPER1(R4),RPERRS+2 ;WILL TELL US SOMETHING
7367 040514 016437 000040 034322      MOV      RPER2(R4),RPERRS+4
7368 040522 016437 000042 034324      MOV      RPER3(R4),RPERRS+6
7369 040530 004037 034714      JSR      RD,DRVINT  ;INIT. THE STATE OF THE DRIVE
7370 040534 000401          BR      1S            ;TAKE ERROR EXIT
7371 040536 000207          RTS      PC          ;RETURN

```



```

7372 040540 005726          1S:   TST      (SP)+      ;POP PC OFF OF THE STACK
7373 040542 000704          BR       SC8        ;PROCESS THE PARITY ERROR
7374 040544 006301          SC13:  ASL      R1        ;SETUP TO ADDRESS WORDS
7375 040546 012761 177777 034430  MOV     #-1,TIMER(R1) ;STOP THE TIMER
7376 040554 006201          ASR     R1          ;
7377 040556 010164 000010  MOV     R1,RPCS2(R4) ;SELECT THE DRIVE
7378 040562 116164 034452 000016  MOVB   ATABIT(R1),RPAS(R4) ;CLEAR THE ATTENTION BIT
7379 040570 032714 004000  BIT     #BIT11,(R4)  ;DRIVE AVAILABLE ?
7380 040574 001006          BNE     1S         ;BR IF AVAILABLE
7381 040576 006301          ASL     R1          ;
7382 040600 012761 023420 034430  MOV     #10000.,TIMER(R1) ;START 10 SEC TIMER AGAIN
7383 040606 006201          ASR     R1          ;
7384 040610 000433          BR      3S         ;EXIT
7385 040612 105761 034356          1S:   TSTB   DPINT(R1) ;INITIALIZING THE DRIVE ?
7386 040616 001424          BEQ     2S         ;BR IF NOT
7387 040620 105061 034356          CLRB   DPINT(R1)  ;CLEAR THE INIT INDICATOR
7388 040624 004037 034714          JSR    RD,DRVINT  ;GO INIT THE DRIVE
7389 040630 000240          NOP                    ;DUMMY PARITY ERROR RETURN
7390 040632 105761 034336          TSTB   DRVSTA(R1) ;DRIVE ONLINE ?
7391 040636 003014          BGT     2S         ;BR IF YES -- START ORDER
7392 040640 005702          TST     R2         ;QUEUE ENTRY FOR THE DRIVE
7393 040642 001416          BEQ     3S         ;BR IF NOT
7394 040644 004737 042640          JSR    PC,GETREQ  ;GET DPB ADDRESS
7395 040650 052762 140000 000016  BIS    #BIT15:BIT14,16(R2) ;INFORM USER THAT DRIVE OFFLINE
7396 040656 004737 042004          JSR    PC,SVRH11 ;SAVE THE REGISTERS
7397 040662 004737 042544          JSR    PC,EMPTYQ ;EMPTY THE REQUEST QUEUE
7398 040666 000404          BR      3S         ;
7399 040670 105061 034366          2S:   CLRB   DPRQS(R1) ;CLEAR THE PORT REQUEST INDICATOR
7400 040674 004737 035544          JSR    PC,OPT     ;START THE PENDING REQUEST
7401 040700 000137 037626          3S:   JMP     SC4     ;PROCESS OTHER DRIVES
7402
7403          ;RPO4/5/6 TIMER ROUTINE
7404          ;CALL
7405          ;
7406          ;
7407          ;
7408 040704 005737 034402          R-TMR: TST     ACTDRV   ;CHECK "ACTDRV & ACTSTR"
7409 040710 001030          BNE     4S         ;IF NON ZERO EXIT
7410 040712 112737 000001 034403  MOVB   #1,ACTSTR  ;SET "ACTSTR"
7411 040720 104412          SAVREG             ;SAVE R0 - R5
7412 040722 005001          CLR    R1         ;START WITH DRIVE 0
7413 040724 005003          CLR    R3         ;
7414 040726 005763 034430          1S:   TST     TIMER(R3) ;IS THE TIMER RUNNING?
7415 040732 002407          BLT     2S         ;BRANCH IF NO
7416 040734 166663 000002 034430  SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
7417 040742 003003          BGT     2S         ;BR IF NO SOFTWARE TIMEOUT
7418 040744 004737 040776          JSR    PC,STO     ;CALL SOFTWARE TIMEOUT ROUTINE
7419 040750 000405          BR      3S         ;GO TO THE EXIT
7420 040752 005201          2S:   INC     R1     ;MOVE TO NEXT DRIVE
7421 040754 005723          TST     (R3)+     ;
7422 040756 022701 000010  CMP     #8.,R1    ;OUT OF DRIVES?
7423 040762 003361          BGT     1S         ;BRANCH IF NO
7424 040764 104413          3S:   RESREG             ;RESTORE R0 - R5
7425 040766 105037 034403          CLRB   ACTSTR    ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
7426 040772 012616          4S:   MOV     (SP)+,(SP) ;ADJUST THE STACK
7427 040774 000207          RTS     PC        ;RETURN
    
```



```

7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439 040776 010146
7440 041000 010346
7441 041002 013704 034464
7442 041006 010164 000010
7443 041012 004037 041432
7444 041016 000012
7445 041020 041320
7446 041022 105726
7447 041024 100477
7448 041026 105761 034356
7449 041032 001074
7450 041034 105761 034366
7451 041040 001071
7452 041042 013702 034376
7453 041046 020137 034450
7454 041052 001402
7455 041054 004737 042640
7456 041060 052762 101000 000016 15:
7457 041066 004737 042004
7458 041072 012764 000040 000010
7459 041100 105061 034326
7460 041104 105061 034404
7461 041110 005001
7462 041112 005003
7463 041114 004037 034714 25:
7464 041120 000477
7465 041122 105761 034326
7466 041126 001414
7467 041130 013702 034376
7468 041134 023701 034450
7469 041140 001402
7470 041142 004737 042640
7471 041146 052762 100400 000016 35:
7472 041154 105061 034326
7473 041160 105061 034404 45:
7474 041164 012763 177777 034430
7475 041172 005723
7476 041174 005201
7477 041176 022701 000010
7478 041202 003344
7479 041204 012737 177777 034450
7480 041212 005037 034376
7481 041216 004737 042466
7482 041222 000500
7483 041224 116405 000016
    
```

SOFTWARE TIMEOUT ROUTINE

NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6 OR GREATER

CALL: STO ;DRIVE NUMBER
 MOV #DRVNUM,R1 ;CALL
 JSR PC,STO
 RETURN

```

STO: MOV R1,-(SP) ;SAVE R1
      MOV R3,-(SP) ;SAVE R3
      MOV RPADR,R4 ;GET ADDRESS OF "RPCS1"
      MOV R1,RPCS2(R4) ;SELECT THE DRIVE
      JSR RD,RD.RP ;READ "DRIVE STATUS REG"
      RPDS1
      STOS
      TSTB (SP)+ ;IS "DRY"=1?
      BMI ST02 ;BR IF YES
      TSTB DPINT(R1) ;TRYING TO INITIALIZE THE DRIVE ?
      BNE ST02 ;BR IF YES
      TSTB DPRQS(R1) ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
      BNE ST02 ;BR IF YES
      MOV TRANSW,R2 ;PICKUP TRANSFER WAIT QUEUE
      CMP R1,DTUM ;TRANSFER UNDERWAY ON THIS DRIVE?
      BEQ 15 ;BRANCH IF YES
      JSR PC,GETREQ ;GET DPB ADDRESS
      BIS #BIT15:BIT09,16(R2) ;SET THE ERROR FLAGS
      JSR PC,SVRH11 ;SAVE RH11/RPO4/5/6 REGISTERS
      MOV #BIT05,RPCS2(R4) ;"INIT" THE MASS BUS
      CLRB DRVACT(R1) ;DRIVE IS IDLE
      CLRB ULDFLG(R1) ;CLEAR THE UNLOAD FLAG
      CLR R1 ;START WITH DRIVE 0
      CLR R3
      JSR RD,DRVINT ;INIT. THIS DRIVE
      BR ST05 ;PARITY ERROR RETURN
      TSTB DRVACT(R1) ;DRIVE IDLE BEFORE THE INIT.?
      BEQ 45 ;YES--BRANCH
      MOV TRANSW,R2 ;GET TRANSFER WAIT QUEUE
      CMP DTUM,R1 ;WAS THERE I/O ON THIS DRIVE?
      BEQ 35 ;YES--BRANCH
      JSR PC,GETREQ ;GET THE DPB POINTER FROM QUEUE
      BIS #BIT15:BIT08,16(R2) ;INFORM USER OF INIT.
      CLRB DRVACT(R1) ;SET DRIVE ACTIVE TO IDLE
      CLRB ULDFLG(R1) ;NO UNLOAD
      MOV #-1,TIMER(R3) ;STOP THE TIMER
      TST (R3)+ ;UPDATE THE INDEX
      INC R1 ;INCREMENT THE DRIVE NUMBER
      CMP #8.,R1 ;LAST DRIVE BEEN CHECKED?
      BGT 25 ;NO--LOOP
      MOV #-1,DTUM ;NO DATA TRANSFERS UNDERWAY
      CLR TRANSW ;CLEAR TRANSFER WAIT QUEUE
      JSR PC,CLRQUE ;CLEAR ALL REQUEST QUEUES
      BR ST09 ;EXIT
      MOVB RPAS(R4),R5 ;READ ATTENTION REG
    
```



```

7484 041230 136105 034452 BITB ATABIT(R1),R5 ;IS ATTENTION FOR THIS DRIVE UP ?
7485 041234 001017 BNE ST03 ;YES--BRANCH
7486 041236 105761 034356 TSTB DPINT(R1) ;TRYING TO INITIALIZE THE DRIVE ?
7487 041242 001031 BNE ST06 ;BR IF YES - DRIVE NOT ONLINE
7488 041244 105761 034366 TSTB DPRQS(R1) ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
7489 041250 001045 BNE ST07 ;BR IF YES - NO RESPONSE TO REQUEST
7490 041252 020137 034450 CMP R1,DTUM ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
7491 041256 001263 BNE ST01 ;BR IF NO
7492 041260 004037 041432 JSR RD,RO.RP ;YES--CHECK "RDY"
7493 041264 000000 RPCS1
7494 041266 041320 ST05
7495 041270 105726 TSTB (SP)+
7496 041272 100255 BPL ST01 ;BR IF "RDY"=0
7497 041274 105761 034356 ST03: TSTB DPINT(R1) ;INITIALIZING THE DRIVE ?
7498 041300 001003 BNE IS ;BR IF INIT PENDING
7499 041302 105761 034366 TSTB DPRQS(R1) ;PORT REQUEST PENDING ?
7500 041305 001446 BEQ ST09 ;BR IF NOT
7501 041310 012763 177777 034430 IS: MOV #-1,TIMER(R3) ;STOP THE TIMER
7502 041316 000442 BR ST09 ;EXIT
7503 041320 004737 036754 ST05: JSR PC,CIB ;GO HANDLE THE PARITY ERROR
7504 041324 000437 BR ST09
7505 041326 105061 034356 ST06: CLRB DPINT(R1) ;CLEAR THE INITIALIZE INDICATOR
7506 041332 105061 034336 CLRB DRVSTA(R1) ;SET UNIT OFFLINE
7507 041336 012763 177777 034430 MOV #-1,TIMER(R3) ;STOP THE TIMER
7508 041344 004737 042640 JSR PC,GETREQ ;GET THE DPB ADDRESS
7509 041350 005702 TST R2 ;REQUEST IN QUEUE ?
7510 041352 001424 BEQ ST09 ;BR IF NOT
7511 041354 052762 140000 000016 BIS #BIT15!BIT14,16(R2) ;INFORM THE USER DRIVE NOT AVAILABLE
7512 041362 000414 BR ST08 ;FINISH
7513 041364 012763 177777 034430 ST07: MOV #-1,TIMER(R3) ;STOP THE TIMER
7514 041372 105061 034366 CLRB DPRQS(R1) ;CLEAR PORT REQUEST INDICATOR
7515 041376 004737 042640 JSR PC,GETREQ ;GET DPB ADDRESS
7516 041402 005702 TST R2 ;QUEUE ENTRY FOR DRIVE ?
7517 041404 001407 BEQ ST09 ;BR IF NONE
7518 041406 012762 100004 000016 MOV #BIT15!BIT2,16(R2) ;INFORM USER OF PORT REQUEST ERROR
7519 041414 004737 042544 ST08: JSR PC,EMPTYQ ;CLEAR THE QUEUE FOR THE DRIVE
7520 041420 004737 042004 JSR PC,SVRH11 ;SAVE THE REGISTERS
7521 041424 012603 ST09: MOV (SP)+,R3 ;RESTORE R3
7522 041426 012601 MOV (SP)+,R1 ;RESTORE R1
7523 041430 000207 RTS PC ;RETURN
7524
7525 ;ROUTINE TO READ A RH11/RPO4/5/6 REGISTER
7526
7527 ;CALL
7528 JSR RD,RO.RP ;GO READ A REGISTER
7529 INDEX ;REG. INDEX FROM BASE
7530 ERRADR ;ERROR ADDRESS--PROCESS ERROR STARTING
7531 ;AT THIS ADDRESS
7532 ;CONTENTS OF REG. IS ON THE STACK
7533
7534 041432 013737 034462 041600 RD.RP: MOV MCPENX,RO.RP2 ;MAX. RETRYS ALLOWED
7535 041440 011646 MOV (SP)-,(SP) ;SAVE RD FOR RETURN
7536 041442 013737 034464 041456 MOV RPADR,RO.ADR ;FORM THE DESIRED ADDRESS
7537 041450 062037 041456 ADD (RO)+,RO.ADR ;USING THE BASE AND THE INDEX
7538 041454 013727 RD.RP1: MOV @((PC)+),(PC)+ ;READ THE DESIRED REGISTER OF THE RPO4
7539 041456 000000 RD.ADR: .WORD 0 ;ADDRESS IS FORMED HERE
    
```



```

7540 041460 000000 RD.WRD: .WORD 0 ;REG. CONTENTS PUT HERE
7541 041462 013766 041460 000002 MOV RD.WRD,2(SP) ;RETURN IT TO THE USER
7542 041470 013746 034464 MOV RPADR,-(SP) ;PUT THE ADDRESS ON THE STACK
7543 041474 062716 000010 ADD #RPCS2,(SP) ;FORM THE ADDRESS OF RPCS2
7544 041500 032736 010000 BIT #BIT12,2(SP)+ ;CHECK THE 'MED' BIT
7545 041504 001037 BNE RD.RP3 ;BR IF DRIVE NON-EXISTENT
7546 041506 017746 172752 MOV #RPADR,-(SP) ;READ RPCS1
7547 041512 032716 020000 BIT #BIT13,(SP) ;DID MCPE SET?
7548 041516 001002 BNE IS ;BRANCH IF YES
7549 041520 022620 CMP (SP)+,(RD)+ ;ADJUST FOR RETURN
7550 041522 000432 BR RD.RP4 ;EXIT
7551 041524 IS: JSR RD,ES.SAV ;SAVE THE ADDRESS IN 'SESCAPE'
7552 041524 004037 042724 ERROR 3 ;REPORT "MCPE" ERROR
7553 041530 104003 TST DTLM ;DATA TRANSFER UNDERWAY?
7554 041532 005737 034450 BMI 2S ;NO--BRANCH
7555 041536 100405 BIT #BIT14,(SP) ;NO--"TRE"=1?
7556 041540 032716 040000 BEQ 2S ;NO--BRANCH
7557 041544 001402 TST (SP)+ ;YES--CLEAN OFF THE STACK AND
7558 041546 005726 BR RD.RP3 ;TAKE THE FATAL ERROR EXIT
7559 041550 000415 BIS #BIT14,(SP) ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
7560 041552 052716 040000 2S: SWAB (SP) ;POSITION BEFORE WRITING
7561 041556 000316 MOV RPADR,3S ;FORM ADDRESS OF HIGH BYTE
7562 041560 013737 034464 041574 INC 3S
7563 041566 005237 041574 MOVB (SP)+,2(PC)+ ;WRITE THE HIGH BYTE OF RPCS1
7564 041572 112637 3S: .WORD 0 ;ADDRESS STORAGE
7565 041574 000000 DEC (PC)+ ;EXCEEDED MAX. RETRYS
7566 041576 005327 RD.RP2: .WORD 3
7567 041600 000003 BGE RD.RP1 ;BRANCH IF NO
7568 041602 002324 RD.RP3: MOV (RD),RD ;FATAL ERROR EXIT
7569 041604 011000 MOV (SP)+,(SP)
7570 041606 012616 RD.RP4: RTS RD
7571 041610 000200
7572
7573 ;ROUTINE TO WRITE A REGISTER
7574 ;CALL
7575 ;
7576 ;
7577 ;
7578 ;
7579 ;
7580 ;
7581 ;
7582 041612 013737 034462 041766 WRT.RP: MOV MCPMX,WRT.R2 ;MAX RETRYS ALLOWED
7583 041620 016637 000002 041700 MOV 2(SP),WRT.WD ;SAVE THE WORD TO WRITE
7584 041626 012616 MOV (SP)+,(SP) ;ADJUST THE STACK
7585 041630 012037 041702 MOV (RD)+,WRT.AD ;GET INDEX OF REGISTER TO BE WRITTEN
7586 041634 001015 BNE IS ;BRANCH IF NOT RPCS1
7587 041636 122737 000150 041700 CMPB #150,WRT.WD ;IS THE COMMAND FOR DATA TRANSFERS?
7588 041644 002411 BLT IS ;YES--DON'T GET THE OLD A16 & A17, & PSEL
7589 041646 004037 041432 JSR RD,RD.RP ;NO---COMBINE A16&A17, & PSEL WITH
7590 041652 000000 RPCS1 ;THE COMMAND BEFORE SENDING IT TO
7591 041654 041774 WRT.R3 ;THE RH11/RPO4
7592 041656 000316 SWAB (SP)
7593 041660 042716 177770 BIC #1C7,(SP)
7594 041664 112637 041701 MOVB (SP)+,WRT.WD+1
7595 041670 063737 034464 041702 1S: ADD RPADR,WRT.AD ;FORM THE ADDRESS OF THE DISK REG.
    
```


7596	041676	012737				WRT.R1: MOV	(PC)+,2(PC)+	:LOAD THE DESIRED REG.
7597	041700	000000				WRT.WD: .WORD	0	:WORD TO WRITE GOES HERE
7598	041702	000000				WRT.AD: .WORD	0	:ADDRESS IS FORMED HERE
7599	041704	013746	034464			MOV	RPADR, -(SP)	:PUT THE ADDRESS ON THE STACK
7600	041710	062716	000010			ADD	#RPCS2, (SP)	:FORM THE ADDRESS OF RPCS2
7601	041714	032736	010000			BIT	#BIT12, 2(SP)+	:CHECK THE 'NED' BIT
7602	041720	001025				BNE	WRT.R3	:BR IF DRIVE NON-EXISTENT
7603	041722	004037	041432			JSR	RD, RD.RP	:CHECK FOR PARITY ERROR ON WRITE
7604	041726	000014				RPER1		
7605	041730	041774				WRT.R3		
7606	041732	032726	000010			BIT	#BIT03, (SP)+	
7607	041736	001420				BEQ	WRT.R4	:BRANCH IF "PAR=0"
7608	041740	016037	177776	041752		MOV	-2(RD), 1\$:PICKUP THE INDEX
7609	041746	004037	041432			JSR	RD, RD.RP	:READ THE REG.
7610	041752	000000				1\$: .WORD	0	:REG. INDEX
7611	041754	041774				WRT.R3		:RETURN TO THIS ADDRESS ON ERROR
7612	041756	004037	042724			JSR	RD, ES.SAV	:SAVE THE ADDRESS IN 'SESCAPE'
7613	041762	104004				ERROR	4	:REPORT THE PARITY ON WRITE ERROR
7614	041764	005327				DEC	(PC)+	:DECREMENT THE ERROR COUNT
7615	041766	000003				WRT.R2: .WORD	3	:RETRY COUNTER
7616	041770	002342				BGE	WRT.R1	:TRY AGAIN IF NOT FINISHED
7617	041772	005726				TST	(SP)+	:CLEAN OFF THE STACK
7618	041774	011000				WRT.R3: MOV	(RD), RD	:TAKE THE "PARITY ON WRITE" ERROR EXIT
7619	041776	000401				BR	WRT.R5	:EXIT
7620	042000	005720				WRT.R4: TST	(RD)+	:ADJUST FOR ERROR FREE EXIT
7621	042002	000200				WRT.R5: RTS	RD	
7622								
7623								:ROUTINE TO SAVE THE RH11/RPO4/5/6 REGISTERS AS PER DPB+14
7624								
7625								
7626								
7627								
7628								
7629	042004	104412				SVRH11: SAVREG		:SAVE RD - R5
7630	042006	005702				TST	R2	:QUEUE ENTRY FOR THE DRIVE ?
7631	042010	001430				BEQ	4\$:BR IF NONE
7632	042012	013704	034464			MOV	RPADR, R4	
7633	042016	111264	000010			MOVB	(R2), RPCS2(R4)	:SELECT DRIVE
7634	042022	016203	000014			MOV	14(R2), R3	:GET THE ERROR TABLE POINTER
7635	042026	001433				BEQ	6\$:EXIT IF NO ADDRESS
7636	042030	005037	042064			CLR	3\$:COUNTER & POINTER
7637	042034	023727	042064	000022	1\$:	CMP	3\$, #RPDB	:REACHED THE BUFFER REGISTER ?
7638	042042	001006				BNE	2\$:BR IF NOT
7639	042044	032764	000200	000010		BIT	#BIT07, RPCS2(R4)	: 'OR' SET ?
7640	042052	001002				BNE	2\$:BR IF SET
7641	042054	005023				CLR	(R3)+	:STORE RPDB AS ZEROES
7642	042056	000405				BR	4\$:CONTINUE
7643	042060	004037	041432		2\$:	JSR	RD, RD.RP	:READ THE SELECTED REGISTER
7644	042064	000000			3\$:	.WORD	0	:REGISTER INDEX
7645	042066	042112				5\$:ERROR RETURN ADDRESS
7646	042070	012623				MOV	(SP)+, (R3)+	:STORE THE REGISTER CONTENTS
7647	042072	023727	042064	000046	4\$:	CMP	3\$, #RPEC2	:REACHED THE END ?
7648	042100	001406				BEQ	6\$:BR IF YES
7649	042102	062737	000002	042064		ADD	#2, 3\$:INCREMENT THE REGISTER INDEX
7650	042110	000751				BR	1\$:CONTINUE READING THE REGISTERS
7651	042112	004737	036654		5\$:	JSR	PC, CI7	:PROCESS THE UNCORRECTABLE PARITY ERROR


```

7652 042116 104413      6S:  RESREG          ;RESTORE R0 - R5
7653 042120 000207      RTS      PC          ;RETURN
7654
7655 ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
7656 ;CALL
7657     MOV      #DRVNUM,R1 ;DRIVE NUMBER TO R1
7658     JSR      PC,SET.IE  ;SET "IE"
7659     RETURN
7660
7661 042122 010446      SET.IE: MOV      R4,-(SP)   ;SAVE R4
7662 042124 013704 034464  MOV      RPADR,R4    ;PICKUP ADDRESS OF RPCS1
7663 042130 010164 000010  MOV      R1,RPCS2(R4) ;SELECT DRIVE
7664 042134 011446      MOV      (R4),-(SP)  ;READ RPCS1
7665 042136 052716 040000  BIS      #BIT14,(SP) ;SET THE "TRE" BIT OF THE WORD READ
7666 042142 000316      SWAB      (SP)      ;ADJUST FOR DATO
7667 042144 112714 000100  MOVB     #BIT06,(R4) ;SET "IE"
7668 042150 032764 010000 000010  BIT      #BIT12,RPCS2(R4) ;IS "NED"=1?
7669 042156 001002      BNE      IS        ;YES--CLEAR "TRE"
7670 042160 005726      TST      (SP)+     ;CLEAN OFF THE STACK
7671 042162 000402      BR       2S
7672 042164 112664 000001  1S:  MOVB     (SP)+,1(R4) ;CLEAR "TRE"
7673 042170 012604      2S:  MOV      (SP)+,R4  ;RESTORE R4
7674 042172 000207      RTS      PC        ;RETURN TO CALLER
7675
7676 ;QUEUE COUNT
7677 042174      000      GCNT: .BYTE    0          ;DRIVE 0
7678 042175      000      .BYTE    0          ;DRIVE 1
7679 042176      000      .BYTE    0          ;DRIVE 2
7680 042177      000      .BYTE    0          ;DRIVE 3
7681 042200      000      .BYTE    0          ;DRIVE 4
7682 042201      000      .BYTE    0          ;DRIVE 5
7683 042202      000      .BYTE    0          ;DRIVE 6
7684 042203      000      .BYTE    0          ;DRIVE 7
7685
7686 ;QUEUE INPUT POINTERS
7687
7688 042204 042266      QINPT: .WORD    QDRV0     ;DRIVE 0
7689 042206 042306      .WORD    QDRV1     ;DRIVE 1
7690 042210 042326      .WORD    QDRV2     ;DRIVE 2
7691 042212 042346      .WORD    QDRV3     ;DRIVE 3
7692 042214 042366      .WORD    QDRV4     ;DRIVE 4
7693 042216 042406      .WORD    QDRV5     ;DRIVE 5
7694 042220 042426      .WORD    QDRV6     ;DRIVE 6
7695 042222 042446      .WORD    QDRV7     ;DRIVE 7
7696
7697 ;QUEUE OUTPUT POINTERS
7698
7699 042224 042266      QOUTPT: .WORD    QDRV0     ;DRIVE 0
7700 042226 042306      .WORD    QDRV1     ;DRIVE 1
7701 042230 042326      .WORD    QDRV2     ;DRIVE 2
7702 042232 042346      .WORD    QDRV3     ;DRIVE 3
7703 042234 042366      .WORD    QDRV4     ;DRIVE 4
7704 042236 042406      .WORD    QDRV5     ;DRIVE 5
7705 042240 042426      .WORD    QDRV6     ;DRIVE 6
7706 042242 042446      .WORD    QDRV7     ;DRIVE 7
7707
    
```



```

7708 042244 042266 QSTART: .WORD QDRV0 ;DRIVE 0 START ADDRESS
7709 042246 042306 QSTOP: .WORD QDRV1 ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
7710 042250 042326 .WORD QDRV2 ;STOP DRIVE 1--START DRIVE 2
7711 042252 042346 .WORD QDRV3 ;STOP DRIVE 2--START DRIVE 3
7712 042254 042366 .WORD QDRV4 ;STOP DRIVE 3--START DRIVE 4
7713 042256 042406 .WORD QDRV5 ;STOP DRIVE 4--START DRIVE 5
7714 042260 042426 .WORD QDRV6 ;STOP DRIVE 5--START DRIVE 6
7715 042262 042446 .WORD QDRV7 ;STOP DRIVE 6--START DRIVE 7
7716 042264 042466 .WORD QTERM ;STOP DRIVE 7
7717
7718 ;DRIVE REQUEST QUEUES
7719
7720 042266 000010 QDRV0: .BLKW 10
7721 042306 000010 QDRV1: .BLKW 10
7722 042326 000010 QDRV2: .BLKW 10
7723 042346 000010 QDRV3: .BLKW 10
7724 042366 000010 QDRV4: .BLKW 10
7725 042406 000010 QDRV5: .BLKW 10
7726 042426 000010 QDRV6: .BLKW 10
7727 042446 000010 QDRV7: .BLKW 10
7728 042466 QTERM=.
7729
7730 ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
7731
7732 ;CALL
7733 ; JSR PC,CLRQUE
7734
7735 042466 104412 CLRQUE: SAVREG ;SAVE R0 - R5
7736 042470 012702 042174 MOV #QCNT,R2 ;ZERO THE QUEUE COUNTS
7737 042474 005022 CLR (R2)+ ;DRIVES 0 & 1
7738 042476 005022 CLR (R2)+ ;DRIVES 2 & 3
7739 042500 005022 CLR (R2)+ ;DRIVES 4 & 5
7740 042502 005022 CLR (R2)+ ;DRIVES 6 & 7
7741 042504 012703 000010 MOV #R,R3 ;MOVE THE STARTING
7742 042510 012701 042244 MOV #QSTART,R1 ;ADDRESS OF THE QUEUE INTO
7743 042514 012122 1S: MOV (R1)+,(R2)+ ;THE QUEUE INPUT POINTER
7744 042516 005303 DEC R3
7745 042520 001375 BNE 1S
7746 042522 012703 000010 MOV #R,R3 ;MOVE THE STARTING ADDRESS
7747 042526 012701 042244 MOV #QSTART,R1 ;OF THE QUEUE INTO THE
7748 042532 012122 2S: MOV (R1)+,(R2)+ ;QUEUE OUTPUT POINTER
7749 042534 005303 DEC R3
7750 042536 001375 BNE 2S
7751 042540 104413 RESREG ;RESTORE R0 - R5
7752 042542 000207 RTS PC
7753
7754 ;EMPTY THE QUEUE SPECIFIED BY R1
7755 ;CALL
7756 ;
7757 ; MOV DRVNUM,R1 ;DRIVE NUMBER TO R1
7758 ; JSR PC,EMPTYQ
7759
7760 042544 105061 042174 EMPTYQ: CLRB QCNT(R1) ;CLEAR NUMBER OF ITEMS IN QUEUE
7761 042550 006301 ASL R1
7762 042552 016161 042204 042224 MOV @INPT(R1),@OUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
7763 042560 006201 ASR R1
    
```



```

7764 042562 000207          RTS      PC
7765
7766          ;ROUTINE TO PUT A REQUEST IN QUEUE
7767          ;CALL
7768          ;
7769          ;MOV      #DRVNUM,R1      ;DRIVE NUMBER
7770          ;MOV      #DPB,R2        ;ADDRESS OF PARAMETER BLOCK
7771          ;JSR      RD,DRVQUE     ;GO PUT REQUEST IN QUEUE
7772          ;RETURN1     ;RETURN HERE IF QUEUE IS FULL
7773          ;RETURN2     ;RETURN HERE IF REQUEST IS IN QUEUE
7774
7775 042564 122761 000010 042174 DRVQUE:  CMPB   #10,QCNT(R1)  ;IS QUEUE FULL?
7776 042572 001421          BEQ     25          ;BR IF YES-TAKE RETURN1
7777 042574 105261 042174          INCB   QCNT(R1)    ;INCREMENT QUEUE COUNT
7778 042600 006301          ASL    R1
7779 042602 010271 042204          MOV    R2,@QINPT(R1) ;PUT THIS REQUEST IN QUEUE
7780 042606 062761 000002 042204          ADD    #2,QINPT(R1) ;UPDATE THE QUEUE POINTER
7781 042614 026161 042204 042246          CMP    QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
7782 042622 001003          BNE   15          ;BRANCH IF NO
7783 042624 016161 042244 042204          MOV    QSTART(R1),QINPT(R1) ;YES--RESET POINTER
7784 042632 006201          15:   ASR    R1
7785 042634 005720          TST   (R0)+      ;TAKE RETURN 2
7786 042636 000200          25:   RTS     R0      ;RETURN TO USER
7787
7788          ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
7789          ;CALL
7790          ;
7791          ;MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
7792          ;JSR      PC,GETREQ      ;GO GET THE REQUEST
7793          ;RETURN      ;R2="DPB" ADDRESS OF THE REQUEST
7794          ;R2=0 IF NO REQUEST IN QUEUE
7795
7796 042640 005002          GETREQ: CLR    R2
7797 042642 105761 042174          TSTB  QCNT(R1)    ;IS THERE ANY REQUEST IN QUEUE?
7798 042646 001404          BEQ   25          ;NO---BRANCH
7799 042650 006301          15:   ASL    R1
7800 042652 017102 042224          MOV    @QOUTPT(R1),R2 ;PICK UP "DPB" POINTER FOR THIS DRIVE
7801 042656 006201          ASR   R1
7802 042660 000207          25:   RTS     PC      ;RETURN TO USER
7803
7804          ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
7805          ;CALL
7806          ;
7807          ;MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
7808          ;JSR      PC,POPGUE     ;CALL TO REMOVE REQUEST
7809          ;RETURN      ;R2=ADDRESS OF DPB REMOVED
7810
7811 042662 105361 042174          POPQUE: DECB  QCNT(R1) ;DECREMENT QUEUE COUNT
7812 042666 006301          ASL   R1
7813 042670 017102 042224          MOV    @QOUTPT(R1),R2 ;GET THE "DPB" POINTER
7814 042674 062761 000002 042224          ADD    #2,QOUTPT(R1) ;UPDATE THE QUEUE POINTER
7815 042702 026161 042224 042246          CMP    QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
7816 042710 001003          BNE   15          ;NO--BRANCH TO EXIT
7817 042712 016161 042244 042224          MOV    QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
7818 042720 006201          15:   ASR   R1
7819 042722 000207          RTS    PC      ;RETURN TO USER
    
```



```

7820
7821
7822
7823
7824
7825
7826
7827
7828
7829 042724 012037 042740
7830 042730 013746 001206
7831 042734 005037 001206
7832 042740 000000
7833 042742 012537 001206
7834 042746 000200
7835
7836
7837
7838
7839 042750 000122
7840 042752 041506 000
7841 042755 114 000103
7842 042760 041506 000047
7843 042764 041514 000047
7844 042770 041511 000
7845 042773 106 000124
7846 042776 052114 000
7847 043001 111 000124
7848 043004 051506 000
7849 043007 114 000123
7850 043012 040520 000124
7851 043016 000075
7852 043020 005015 047503 052116
7853 043026 047522 020114 053523
7854 043034 052111 044103 051505
7855 043042 000075
7856
7857 043044 027440 000040
7858 043050 047125 052111 051440
7859 043056 040524 052524 035123
7860 043064 005015 000012
7861 043070 051104 053111 000105
7862 043076 047440 043106 044514
7863 043104 042516 000
7864 043107 040 047117 044514
7865 043114 042516 000
7866 043117 040 047516 020124
7867 043124 051120 051505 047105
7868 043132 000124
7869 043134 052440 051516 043101
7870 043142 000105
7871 043144 047040 052117 051040
7872 043152 030120 027464 027465
7873 043160 000066
7874 043162 050122 032060 000
7875 043167 122 030120 000065

```

```

:ROUTINE TO SAVE THE CONTENTS OF 'SESCAPE' WHEN THE DRIVER
:REPORTS AN ERROR DIRECTLY.
:CALL
:      JSR      RD,ES.SAV
:      ERROR   N
:      RETURN
:THE ERROR CALL
:THE RETURN IS PAST THE ERROR CALL
ES.SAV: MOV      (RD)+,IS
:      MOV      SESCOPE,-(SP)
:      CLR      SESCOPE
:GET THE ERROR CALL
:SAVE THE ADDRESS IN 'SESCAPE'
:CLEAR THE ESCAPE RETURN
:THE ERROR CALL IS MOVED HERE
:RESTORE THE ESCAPE ADDRESS
:RETURN
IS:     .WORD   0
:      MOV      (SP)+,SESCAPE
:      RTS      RD

```

.SBTTL ASCIZ MESSAGES

```

MSG.R:  .ASCIZ  /R/
MSG.FC:  .ASCIZ  /FC/
MSG.LC:  .ASCIZ  /LC/
MSG.FC:  .ASCIZ  /FC'/
MSG.LC:  .ASCIZ  /LC'/
MSG.IC:  .ASCIZ  /IC/
MSG.FT:  .ASCIZ  /FT/
MSG.LT:  .ASCIZ  /LT/
MSG.IT:  .ASCIZ  /IT/
MSG.FS:  .ASCIZ  /FS/
MSG.LS:  .ASCIZ  /LS/
MSG.PAT: .ASCIZ  /PAT/
MSG.EQ:  .ASCIZ  /=/
MSG.CS:  .ASCIZ  <CR><LF>/CONTROL SWITCHES=/
SLASH:  .ASCIZ  @ / @
SYSTAT: .ASCIZ  /UNIT STATUS:/(CR)<LF><LF>
UNMSG:  .ASCIZ  /DRIVE/
UNTOFF: .ASCIZ  / OFFLINE/
UNTON:  .ASCIZ  / ONLINE/
NOTPRS: .ASCIZ  / NOT PRESENT/
NOTSAF: .ASCIZ  / UNSAFE/
NOTIP:  .ASCIZ  @ NOT RPO4/5/6@
RPO4:   .ASCIZ  /RPO4/
RPO5:   .ASCIZ  /RPO5/

```


7876	043174	050122	033060	000	RPO6: .ASCIZ /RPO6/
7877	043201	015	042012	044522	DRIVES: .ASCIZ <CR><LF>/DRIVE(S) TO BE TESTED /
7878	043206	042526	051450	020051	
7879	043214	047524	041040	020105	
7880	043222	042524	052123	042105	
7881	043230	000040			
7882	043232	047516	042516	000	NONE: .ASCIZ /NONE/
7883	043237	054	000		COMMA: .ASCIZ / /
7884	043241	015	047012	020117	NOCLOCK: .ASCIZ <CR><LF>/NO KMI1-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED/
7885	043246	053513	030461	050055	
7886	043254	041440	047514	045503	
7887	043262	020054	044524	044515	
7888	043270	043516	052040	051505	
7889	043276	051524	053440	046111	
7890	043304	020114	047516	020124	
7891	043312	042502	050040	051105	
7892	043320	047506	046522	042105	
7893	043326	000			
7894	043327	015	005012	042524	TESTING: .ASCIZ <CR><LF><LF>/TESTING DRIVE /
7895	043334	052123	047111	020107	
7896	043342	051104	053111	020105	
7897	043350	000			
7898	043351	123	051105	040511	SERIAL: .ASCIZ /SERIAL NUMBER /
7899	043356	020114	052516	041115	
7900	043364	051105	000040		
7901					
7902	043370	005015	051012	052117	MSG7: .ASCIZ <CR><LF><LF>/ROTATIONAL SPEED TIMES/
7903	043376	052101	047511	040516	
7904	043404	020114	050123	042505	
7905	043412	020104	044524	042515	
7906	043420	000123			
7907	043422	005015	047412	042516	MSG10A: .ASCIZ <CR><LF><LF>/ONE CYLINDER SEEK TIMES/<CR><LF>/ * FORWARD/
7908	043430	041440	046131	047111	
7909	043436	042504	020122	047523	
7910	043444	045505	052040	047511	
7911	043452	051505	005015	023040	
7912	043460	043040	051117	047527	
7913	043466	042122	000		
7914	043471	015	020012	020052	MSG10B: .ASCIZ <CR><LF>/ * REVERSE/
7915	043476	042522	042526	051522	
7916	043504	000105			
7917	043506	005015	040412	041503	MSG11A: .ASCIZ <CR><LF><LF>/ACCESS TIME MEASUREMENT/<CR><LF>/ * FORWARD/
7918	043514	051505	020123	044524	
7919	043522	042515	046440	040505	
7920	043530	052523	046522	047105	
7921	043536	006524	020012	020052	
7922	043544	047506	053522	051101	
7923	043552	000104			
7924	043554	005015	025040	051040	MSG11B: .ASCIZ <CR><LF>/ * REVERSE/
7925	043562	053105	051105	042523	
7926	043570	000			
7927	043571	015	005012	040515	MSG12A: .ASCIZ <CR><LF><LF>/MAXIMUM SEEK TIMES/<CR><LF>/ * FORWARD/
7928	043576	044530	052515	020115	
7929	043604	042523	045505	052040	
7930	043612	046511	051505	005015	
7931	043620	025040	043040	051117	

7932	043626	040527	042122	000	
7933	043633	015	020012	020052	MSG128: .ASCIZ <CR><LF>/ * REVERSE/
7934	043640	042522	042526	051522	
7935	043646	000105			
7936					
7937	043650	005015	044515	036516	MSGMIN: .ASCIZ <CR><LF>/MIN=/ /
7938	043656	000			
7939	043657	015	046412	054101	MSGMAX: .ASCIZ <CR><LF>/MAX=/ /
7940	043664	000075			
7941	043666	005015	053101	036507	MSGAVG: .ASCIZ <CR><LF>/AVG=/ /
7942	043674	000			
7943	043675	060	052440	000123	MSGOUS: .ASCIZ /0 US/ /
7944	043702	041040	046105	053517	MBELOW: .ASCIZ / BELOW THE MINIMUM OF / /
7945	043710	052040	042510	046440	
7946	043716	047111	046511	046525	
7947	043724	047440	020106	000	
7948	043731	040	041101	053117	MABOVE: .ASCIZ / ABOVE THE MAXIMUM OF / /
7949	043736	020105	044124	020105	
7950	043744	040515	044530	052515	
7951	043752	020115	043117	000040	
7952	043760	051440	042505	051513	MSGNUM: .ASCIZ / SEEKS TIMED/ /
7953	043766	052040	046511	042105	
7954	043774	000			
7955	043775	040	047516	020124	MSGNON: .ASCIZ / NOT TIMED/ /
7956	044002	044524	042515	000104	
7957	044010	020040	000		MSG.SP: .ASCIZ / / ;TWO (2) SPACES
7958					
7959					.SBTTL ERROR HEADER (EM) MESSAGES
7960					
7961	044013	122	030510	020061	EM1: .ASCIZ /RH11 INTERRUPT OCCURRED (RPAS = 0)/
7962	044020	047111	042524	051122	
7963	044026	050125	020124	041517	
7964	044034	052503	051122	042105	
7965	044042	024040	050122	051501	
7966	044050	036440	030040	000051	
7967	044056	047125	054105	042520	EM2: .ASCIZ /UNEXPECTED ATTENTION OCCURRED/ /
7968	044064	052103	042105	040440	
7969	044072	052124	047105	044524	
7970	044100	047117	047440	041503	
7971	044106	051125	042522	000104	
7972	044114	040515	051523	052502	EM3: .ASCIZ /MASSBUS PARITY ERROR(MCPE=1)/ /
7973	044122	020123	040520	044522	
7974	044130	054524	042440	051122	
7975	044136	051117	046450	050103	
7976	044144	036505	024461	000	
7977	044151	115	051501	041123	EM4: .ASCIZ /MASSBUS PARITY ERROR(PAR=1)/ /
7978	044156	051525	050040	051101	
7979	044164	052111	020131	051105	
7980	044172	047522	024122	040520	
7981	044200	036522	024461	000	
7982	044205	101	042104	042522	EM5: .ASCIZ /ADDRESS PLUG CHANGE BIT SET/ /
7983	044212	051523	050040	052514	
7984	044220	020107	044103	047101	
7985	044226	042507	041040	052111	
7986	044234	051440	052105	000	
7987	044241	122	030510	027461	EM10: .ASCIZ "RH11/RP04/5/6 FAILED TO RESPOND TO ADDRESSING"

7988	044246	050122	032060	032457
7989	044254	033057	043040	044501
7990	044262	042514	020104	047524
7991	044270	051040	051505	047520
7992	044276	042116	052040	020117
7993	044304	042101	051104	051505
7994	044312	044523	043516	000
7995	044317	104	044522	042526
7996	044324	051440	046105	041505
7997	044332	042524	020104	051511
7998	044340	047040	052117	047440
7999	044346	046116	047111	000105
8000	044354	046511	051120	050117
8001	044362	051105	044040	040505
8002	044370	042504	020122	040504
8003	044376	040524	000	
8004	044401	104	052101	020101
8005	044406	047503	050115	051101
8006	044414	020105	040506	046111
8007	044422	051125	000105	
8008	044426	044504	045523	042440
8009	044434	051122	051117	044440
8010	044442	020116	044524	044515
8011	044450	043516	052040	051505
8012	044456	000124		
8013	044460	046103	041517	020113
8014	044466	045450	030527	026461
8015	044474	024520	047440	042526
8016	044502	043122	047514	020127
8017	044510	047111	052040	046511
8018	044516	047111	020107	042524
8019	044524	052123	000	
8020	044527	104	051511	020113
8021	044534	051105	047522	020122
8022	044542	052504	044522	043516
8023	044550	051440	042505	000113
8024	044556	042523	045505	047040
8025	044564	052117	041440	046517
8026	044572	046120	052105	020105
8027	044600	044527	044124	047111
8028	044606	030440	030062	046440
8029	044614	000123		
8030	044616	044122	030461	051057
8031	044624	030120	027464	027465
8032	044632	020066	051105	047522
8033	044640	000122		
8034	044642	040506	040524	020114
8035	044650	051127	052111	020105
8036	044656	044103	041505	020113
8037	044664	051105	047522	000122
8038				
8039				
8040				
8041	044672	043117	046106	047111
8042	044700	020105	051117	052440
8043	044706	051516	043101	020105

EM11: .ASCIZ /DRIVE SELECTED IS NOT ONLINE/

EM12: .ASCIZ /IMPROPER HEADER DATA/

EM13: .ASCIZ /DATA COMPARE FAILURE/

EM17: .ASCIZ /DISK ERROR IN TIMING TEST/

EM20: .ASCIZ /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/

EM23: .ASCIZ /DISK ERROR DURING SEEK/

EM24: .ASCIZ /SEEK NOT COMPLETE WITHIN 120 MS/

EM41: .ASCIZ "RH11/RPO4/5/6 ERROR"

EM46: .ASCIZ /FATAL WRITE CHECK ERROR/

.SBTTL STATUS/ERROR INDICATOR MESSAGES

MSG814: .ASCIZ /OFFLINE OR UNSAFE DRIVE REQUESTED/

8044	044714	051104	053111	020105	
8045	044722	042522	052521	051505	
8046	044730	042524	000104		
8047	044734	047125	047514	042101	MSG813: .ASCIZ /UNLOADED DRIVE REQUESTED/
8048	044742	042105	042040	044522	
8049	044750	042525	051040	050505	
8050	044756	042525	052123	042105	
8051	044764	000			
8052	044765	120	051105	044523	MSG812: .ASCIZ /PERSISTENT UNSAFE/
8053	044772	052123	047105	020124	
8054	045000	047125	040523	042506	
8055	045006	000			
8056	045007	120	051101	052111	MSG811: .ASCIZ /PARITY ERROR OCCURRED/
8057	045014	020131	051105	047522	
8058	045022	020122	041517	052503	
8059	045030	051122	042105	000	
8060	045035	106	052101	046101	MSG810: .ASCIZ /FATAL PARITY ERROR/
8061	045042	050040	051101	052111	
8062	045050	020131	051105	047522	
8063	045056	000122			
8064	045060	047523	052106	040527	MSG809: .ASCIZ /SOFTWARE TIMEOUT ON THIS DRIVE/
8065	045066	042522	052040	046511	
8066	045074	047505	052125	047440	
8067	045102	020116	044124	051511	
8068	045110	042040	044522	042526	
8069	045116	000			
8070	045117	123	043117	053524	MSG808: .ASCIZ /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
8071	045124	051101	020105	044524	
8072	045132	042515	052517	020124	
8073	045140	047117	040440	047516	
8074	045146	044124	051105	042040	
8075	045154	044522	042526	000	
8076	045161	105	051122	051117	MSG806: .ASCIZ "ERROR OCCURRED DURING I/O OPERATION"
8077	045166	047440	041503	051125	
8078	045174	042522	020104	052504	
8079	045202	044522	043516	044440	
8080	045210	047457	047440	042520	
8081	045216	040522	044524	047117	
8082	045224	000			
8083	045225	105	051122	051117	MSG805: .ASCIZ "ERROR OCCURRED DURING NON-I/O OPERATION"
8084	045232	047440	041503	051125	
8085	045240	042522	020104	052504	
8086	045246	044522	043516	047040	
8087	045254	047117	044455	047457	
8088	045262	047440	042520	040522	
8089	045270	044524	047117	000	
8090	045275	125	051516	043101	MSG804: .ASCIZ /UNSAFE OCCURRED/
8091	045302	020105	041517	052503	
8092	045310	051122	042105	000	
8093	045315	101	052125	046517	MSG803: .ASCIZ /AUTOMATIC RECALIBRATE SEQUENCE OCCURRED/
8094	045322	052101	041511	051040	
8095	045330	041505	046101	041111	
8096	045336	040522	042524	051440	
8097	045344	050505	042525	041516	
8098	045352	020105	041517	052503	
8099	045360	051122	042105	000	

8100	045365	104	044522	042526
8101	045372	044040	051501	047040
8102	045400	052117	051040	051505
8103	045406	047520	042116	042105
8104	045414	052040	020117	047520
8105	045422	052122	051040	050505
8106	045430	042525	052123	000
8107	045435	104	044522	042526
8108	045442	044040	051501	041040
8109	045450	041505	046517	020105
8110	045456	047516	026516	054105
8111	045464	051511	042524	052116
8112	045472	000		
8113				
8114				
8115				
8116	045473	105	051122	050040
8117	045500	020103	051040	040520
8118	045506	000123		
8119	045510	051105	020122	041520
8120	045516	020040	051104	053111
8121	045524	020105	020040	050122
8122	045532	051501	020040	020040
8123	045540	050122	051504	020061
8124	045546	020040	050122	051105
8125	045554	020061	020040	050122
8126	045562	051105	020062	020040
8127	045570	050122	051105	000063
8128	045576	042524	052123	020040
8129	045604	020040	051105	020122
8130	045612	041520	020040	042101
8131	045620	051104	051505	020123
8132	045626	040504	040524	000
8133	045633	124	051505	020124
8134	045640	020040	042440	051122
8135	045646	050040	020103	040440
8136	045654	042104	042522	051523
8137	045662	043440	042104	052101
8138	045670	020101	041040	042104
8139	045676	052101	000101	
8140	045702	050122	051503	020061
8141	045710	020040	051105	020122
8142	045716	041520	000	
8143	045721	104	044522	042526
8144	045726	020040	042440	051122
8145	045734	050040	000103	
8146	045740	042524	052123	020040
8147	045746	020040	051105	020122
8148	045754	041520	020040	051524
8149	045762	020124	041520	020040
8150	045770	051104	053111	020105
8151	045776	020040	054503	047114
8152	046004	051104	020040	051124
8153	046012	041501	020113	020040
8154	046020	042523	052103	051117
8155	046026	000		

MSG802: .ASCIZ /DRIVE HAS NOT RESPONDED TO PORT REQUEST/

MSG801: .ASCIZ /DRIVE HAS BECOME NON-EXISTENT/

.SBTTL DATA HEADER (DT) MESSAGES

DH1: .ASCIZ /ERR PC RPAS/

DH2: .ASCIZ /ERR PC DRIVE RPAS RPDS1 RPER1 RPER2 RPER3/

DH3: .ASCIZ /TEST ERR PC ADDRESS DATA/

DH4: .ASCIZ /TEST ERR PC ADDRESS GDDATA BDDATA/

DH10: .ASCIZ /RPCS1 ERR PC/

DH11: .ASCIZ /DRIVE ERR PC/

DH12: .ASCIZ /TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR/

8268	047140	001366	001116		DT10:	.WORD	RH.ADR,SERRPC
8269	047144	001166	001116		DT11:	.WORD	SREG2,SERRPC
8270	047150	001176	001116	001162	DT12:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8271	047156	001254	001270	001274			
8272	047164	001272					
8273	047166	001270	001274	001272	DT12A:	.WORD	CYL.DS,TRK.DS,SEC.DS,CYL.RD,TRK.RD,SEC.RD
8274	047174	001262	001264	001266			
8275	047202	001176	001116	001162	DT13:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8276	047210	001254	001270	001274			
8277	047216	001272					
8278	047220	001124	001126	001172	DT13A:	.WORD	SGDDAT,SBDDAT,SREG4,SGDADR,SBDAADR
8279	047226	001120	001122				
8280	047232	001176	001116	001254	DT17:	.WORD	STMPD,SERRPC,CHKDRV,RP.REG,RP.REG+12,RP.REG+14,RP.REG+40,RP.REG+42
8281	047240	004204	004216	004220			
8282	047246	004244	004246				
8283	047252	001176	001116	001162	DT21:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,CYL.DS,TRK.DS
8284	047260	001254	001270	001274			
8285	047266	001164	001126	001172	DT21A:	.WORD	SREG1,SBDDAT,SREG4,SREG1
8286	047274	001164					
8287	047276	001176	001116	001254	DT23:	.WORD	STMPD,SERRPC,CHKDRV,CYL.DS,RP.REG,RP.REG+10,RP.REG+12
8288	047304	001270	004204	004214			
8289	047312	004216					
8290	047314	004220	004244	004246	DT23A:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+34,RP.REG+36
8291	047322	004240	004242				
8292	047326	001176	001116	001162	DT41:	.WORD	STMPD,SERRPC,SREGO,CHKDRV
8293	047334	001254					
8294	047336	001176	001116	001162	DT42:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,RP.REG,RP.REG+10,RP.REG+12
8295	047344	001254	004204	004214			
8296	047352	004216					
8297	047354	001176	001116	001162	DT43:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,RP.REG,RP.REG+10,RP.REG+12
8298	047362	001254	004204	004214			
8299	047370	004216					
8300	047372	004220	004244	004246	DT43A:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42
8301	047400	001176	001116	001162	DT44:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8302	047406	001254	001270	001274			
8303	047414	001272					
8304	047416	004204	004214	004216	DT44A:	.WORD	RP.REG,RP.REG+10,RP.REG+12,RP.REG+36,RP.REG+34,RP.REG+06
8305	047424	004242	004240	004212			
8306	047432	004220	004244	004246	DT44B:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42
8307	047440	001176	001116	001162	DT45:	.WORD	STMPD,SERRPC,SREGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8308	047446	001254	001270	001274			
8309	047454	001272					
8310	047456	004204	004214	004216	DT45A:	.WORD	RP.REG,RP.REG+10,RP.REG+12,RP.REG+36,RP.REG+34,RP.REG+06
8311	047464	004242	004240	004212			
8312	047472	004220	004244	004246	DT45B:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+2,RP.REG+4,RP.REG+22
8313	047500	004206	004210	004226			

8314
8315 .SBTTL DATA FORMAT (DF) TABLE

8316					DF1:	.WORD	1	; NUMBER OF DATA HEADERS ; NUMBER OF WORDS IN DATA TABLE ; ALL 3 NUMBERS ARE OCTAL
8317	047506	000001			.BYTE	2		
8318	047510	002			.BYTE	0		
8319	047511	000						
8320								
8321	047512	000001			DF2:	.WORD	1	
8322	047514	007			.BYTE	7		
8323	047515	000			.BYTE	0		

8324				
8325	047516	000001	DF3:	.WORD 1
8326	047520	004		.BYTE 4
8327	047521	000		.BYTE 0
8328				
8329	047522	000001	DF4:	.WORD 1
8330	047524	005		.BYTE 5
8331	047525	000		.BYTE 0
8332				
8333	047526	000001	DF10:	.WORD 1
8334	047530	002		.BYTE 2
8335	047531	000		.BYTE 0
8336				
8337	047532	000001	DF11:	.WORD 1
8338	047534	002		.BYTE 2
8339	047535	000		.BYTE 0
8340				
8341	047536	000002	DF12:	.WORD 2
8342	047540	007		.BYTE 7
8343	047541	160		.BYTE 160
8344	047542	046027		.WORD DH12A
8345	047544	006		.BYTE 6
8346	047545	000		.BYTE 0
8347				
8348	047546	000002	DF13:	.WORD 2
8349	047550	007		.BYTE 7
8350	047551	160		.BYTE 160
8351	047552	046106		.WORD DH13A
8352	047554	005		.BYTE 5
8353	047555	004		.BYTE 4
8354				
8355	047556	000000	DF14:	.WORD 0
8356	047560	005		.BYTE 5
8357	047561	004		.BYTE 4
8358				
8359	047562	000001	DF17:	.WORD 1
8360	047564	010		.BYTE 108
8361	047565	000		.BYTE 0
8362				
8363	047566	000002	DF21:	.WORD 2
8364	047570	006		.BYTE 6
8365	047571	060		.BYTE 60
8366	047572	046330		.WORD DH21A
8367	047574	004		.BYTE 4
8368	047575	014		.BYTE 14
8369				
8370	047576	000000	DF22:	.WORD 0
8371	047600	004		.BYTE 4
8372	047601	014		.BYTE 14
8373				
8374	047602	000002	DF23:	.WORD 2
8375	047604	007		.BYTE 7
8376	047605	010		.BYTE 10
8377	047606	046455		.WORD DH23A
8378	047610	005		.BYTE 5
8379	047611	000		.BYTE 0

;2 DH'S TO BE TYPED
;7 DATA WORDS FOLLOW THE 1ST DH
;WORDS 1-4 ARE OCTAL 5-7 ARE DECIMAL
;ADDRESS OF 2ND DH
;6 DATA WORDS FOLLOW THE 2ND DH
;ALL WORDS ARE OCTAL

;WORD 3 IS DECIMAL

;WORD 3 IS DECIMAL

;WORD 4 IS DECIMAL

8380									
8381									
8382	047612	000001			DF41:	.WORD	1		
8383	047614	004				.BYTE	4		
8384	047615	000				.BYTE	0		
8385									
8386	047616	000001			DF42:	.WORD	1		
8387	047620	007				.BYTE	7		
8388	047621	000				.BYTE	0		
8389									
8390	047622	000002			DF43:	.WORD	2		
8391	047624	007				.BYTE	7		
8392	047625	000				.BYTE	0		
8393	047626	046646				.WORD	DM43A		
8394	047630	003				.BYTE	3		
8395	047631	000				.BYTE	0		
8396									
8397	047632	000003			DF44:	.WORD	3		
8398	047634	007				.BYTE	7		
8399	047635	160				.BYTE	160		
8400	047636	046674				.WORD	DM44A		
8401	047640	006				.BYTE	6		
8402	047641	000				.BYTE	0		
8403	047642	046751				.WORD	DM44B		
8404	047644	003				.BYTE	3		
8405	047645	300				.BYTE	0		
8406									
8407	047646	000003			DF45:	.WORD	3		
8408	047650	007				.BYTE	7		
8409	047651	160				.BYTE	160		
8410	047652	046674				.WORD	DM44A		
8411	047654	006				.BYTE	6		
8412	047655	000				.BYTE	0		
8413	047656	046777				.WORD	DM45A		
8414	047660	006				.BYTE	6		
8415	047661	000				.BYTE	0		
8416									
8417									
8418		047662							
8419									
8420	047662	005015	046412	044501					
8421	047670	042116	041505	030455	TITLE:	.ASCII	<CR><LF><LF>/MAINDEC-11-DZRJA-A/<CR><LF>		
8422	047676	026461	055104	045122					
8423	047704	026501	006501	012					
8424	047711	122	030120	027464		.ASCIZ	3RPO4/5/6 MECHANICAL & READ-WRITE TEST<CR><LF><LF>		
8425	047716	027465	020066	042515					
8426	047724	044103	047101	041511					
8427	047732	046101	023040	051040					
8428	047740	040505	026504	051127					
8429	047746	052111	020105	042524					
8430	047754	052123	005015	000012					
8431	047762	005015	047524	052040	LOADRV:	.ASCII	<CR><LF>/TO TEST DRIVE 0 REPLACE THE 'XXDP' PACK ON DRIVE 0/<CR><LF>		
8432	047770	051505	020124	051104					
8433	047776	053111	020105	020060					
8434	050004	042522	046120	041501					
8435	050012	020105	044124	020105					

8436	050020	054047	042130	023520
8437	050026	050040	041501	020113
8438	050034	047117	042040	044522
8439	050042	042526	030040	005015
8440	050050	044527	044124	040440
8441	050056	047516	044124	051105
8442	050064	050040	041501	026113
8443	050072	041440	042514	051101
8444	050100	046440	046505	051117
8445	050106	020131	047514	040503
8446	050114	044524	047117	032040
8447	050122	026060	040440	042116
8448	050130	051040	051505	040524
8449	050136	052122	005015	
8450	050142	044124	020105	051120
8451	050150	043517	040522	006515
8452	050156	000012		
8453	050160	005015	054523	052123
8454	050166	046505	044040	051501
8455	050174	030440	045466	046440
8456	050202	046505	051117	026131
8457	050210	023440	054130	050104
8458	050216	020047	047514	042101
8459	050224	051105	053440	046111
8460	050232	020114	042502	047440
8461	050240	042526	053522	044522
8462	050246	052124	047105	005015
8463	050254	000012		
8464				
8465				
8466				
8467				
8468				
8469				
8470				
8471				
8472				
8473				
8474				
8475	050256	010046		
8476	050260	010146		
8477	050262	013746	000004	
8478	050266	013746	000006	
8479	050272	010600		
8480				
8481	050274	104400		
8482	050276	012637	000006	
8483	050302	012737	050322	000004
8484	050310	012701	020000	
8485	050314	005711		
8486	050316	005721		
8487	050320	000775		
8488	050322	162701	000002	
8489	050326	010006		
8490	050330	012637	000006	
8491	050334	012637	000004	

.ASCII /WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40, AND RESTART/<CR><LF>

.ASCIZ /THE PROGRAM/<CR><LF>

NLOAD: .ASCIZ <CR><LF>/SYSTEM HAS 16K MEMORY, 'XXDP' LOADER WILL BE OVERWRITTEN/<CR><L

.EVEN

.SBTTL ROUTINE TO SIZE MEMORY

::*****

::CALL:

:: JSR PC,SSIZE

:: RETURN

::SLSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION

SSIZE: MOV RO,-(SP) ;:SAVE RO ON THE STACK

MOV R1,-(SP) ;:SAVE R1 ON THE STACK

MOV @ERRVEC,-(SP) ;:SAVE PRESENT ERROR VECTOR PS & PC

MOV @ERRVEC+2,-(SP)

MOV SP,RO ;:SAVE THE STACK POINTER

::SET THE ERRVEC PS TO THE PRESENT PS

TRAP ;:PUSH GLD PSW AND PC ON STACK

MOV (SP)+,@ERRVEC+2 ;:SAVE THE PSW IN @ERRVEC+2

MOV @20000,@ERRVEC ;:SET FOR TIMEOUT

MOV @20000,R1 ;:FIRST ADDRESS

1\$: TST (R1) ;:TEST THIS ADDRESS

TST (R1)+ ;:STEP TO NEXT ADDRESS

BR 1\$;:TRY ANOTHER

2\$: SUB @2,R1 ;:DROP BACK

MOV RO,SP ;:RESTORE THE STACK

MOV (SP)+,@ERRVEC+2 ;:RESTORE ERROR VECTOR

MOV (SP)+,@ERRVEC


```

8492 050340 010137 050352          MOV    R1,SLSTAD          ;;LAST ADDRESS
8493 050344 012601                MOV    (SP)+,R1          ;;RESTORE R1
8494 050346 012600                MOV    (SP)+,R0          ;;RESTORE R0
8495 050350 000207                RTS    PC
8496 050352 000000                SLSTAD: .WORD    0          ;;CONTAINS THE LAST ADDRESS
8497
8498
8499
8500
8501
8502
8503
8504
8505
8506
8507
8508
8509
8510 050354 005737 001226                GETADR: TST    @#BUSADR          ;;INPUT FROM TTY REQUESTED?
8511 050360 001427                BEQ    7$                    ;;NO--BRANCH
8512 050362 005037 001226                CLR    @#BUSADR          ;;YES--CLEAR THE REQUEST FLAG
8513 050366 012700 001366                1$:  MOV    @#RH.ADR,R0          ;;FIRST ADDRESS
8514 050372 012703 050532                MOV    @#RPCS1,R3          ;;RPCS1=
8515 050376 011004                MOV    (R0),R4            ;;PRESENT RPCS1 ADDRESS
8516 050400 004037 031466                JSR    R0,@#GETNUM          ;;GET NEW RPCS1
8517 050404 000402                BR     2$                    ;;COMMA
8518 050406 000767                BR     1$                    ;;PERIOD
8519 050410 000412                BR     5$                    ;;DOUBLE PERIOD
8520 050412 010420                2$:  MOV    R4,(R0)+          ;;SAVE NEW RPCS1
8521 050414 012703 050543                MOV    @#RHVEC,R3          ;;RHVEC=
8522 050420 011004                MOV    (R0),R4            ;;PRESENT RH11 VECTOR ADDRESS
8523 050422 004037 031466                JSR    R0,@#GETNUM          ;;GET NEW RHVEC
8524 050426 000402                BR     3$                    ;;COMMA
8525 050430 000756                BR     1$                    ;;PERIOD
8526 050432 000401                BR     5$                    ;;DOUBLE PERIOD
8527 050434 010420                3$:  MOV    R4,(R0)+          ;;SAVE NEW RHVEC
8528 050436 010410                5$:  MOV    R4,(R0)            ;;SAVE INPUT
8529 050440 013701 000004                7$:  MOV    @#ERRVEC,R1          ;;SAVE THE ERROR VECTOR
8530 050444 012737 050500 000004                MOV    @#ERRVEC,R0          ;;SETUP FOR TRAP
8531 050452 005777 130710                TST    @#RH.ADR          ;;CHECK FOR RH11/RPO4
8532 050456 010137 000004                MOV    R1,@#ERRVEC          ;;RESTORE ERROR VECTOR
8533 050462 012700 001366                MOV    @#RH.ADR,R0          ;;FIRST ADDRESS OF NEW PARAMETERS
8534 050466 012701 034464                MOV    @#PADR,R1           ;;FIRST ADDRESS OF WHERE TO PUT THEM
8535 050472 012021                MOV    (R0)+,(R1)+          ;;BUS ADDRESS
8536 050474 012021                MOV    (R0)+,(R1)+          ;;VECTOR ADDRESS
8537 050476 000207                RTS    PC                    ;;RETURN
8538 050500 010137 000004                8$:  MOV    R1,@#ERRVEC          ;;RESTORE ERROR VECTOR
8539 050504 022626                CMP    (SP)+,(SP)+          ;;CLEAN OFF THE STACK
8540 050506 104010                ERROR 10                    ;;REPORT THE ERROR
8541 050510 005737 000042                TST    @#42                ;;IS THERE A MONITOR?
8542 050514 001724                BEQ    1$                    ;;NO--GO ASK FOR ADDRESS
8543 050516 005037 001232                CLR    @#DRVSEL          ;;YES--NO DRIVES SELECTED
8544 050522 005037 017532                CLR    @#SEOPCT          ;;NO PASSES
8545 050526 000137 017356                JMP    @#SEOP              ;;GO TO END OF PROGRAM
8546
8547 050532 005015 050122 051503 MRPCS1: .ASCIZ <CR><LF>/RPCS1=/>

```


E13

MD-11-DZRJA-AA RPO4/5/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 161
DZRJA.014 19-APR-76 00:00 GETADR - GET BUS ADDRESS AND VECTOR ADDRESS

8548	050540	036461	000	
8549	050543	015	051012	053110 MRHVEC: .ASCIZ <CR><LF>/RHVEC=/ .END
8550	050550	041505	000075	
8551		000001		

L14

CKCHR	1705#	5809	5859	6086	6123	6163	6326	6339	6350	6401					
CKDIG	1705#	5867	6105												
CKNUM	1705#	5817	6138	6171											
COMMEN	198#	1960	1995	2641	3050										
COMND	1705#	1929	2032	2361	2461	3501	3506								
DO	1705#	1928	1930	2039	2040	2075	2076	2112	2120	2154	2159	2200	2206	2222	2228
	2276	2278	2322	2324	2326	2328	2330	2332	2383	2404	2626	2628	3502	3507	5141
DOOTA	1705#	3101	3105	3110	3120	3127	3173	3184	3200	3206	3346	3352	3362	3423	3444
	3450	3458	3464	3472											
DRV. IN	1705#	2748	2842	2941	3039										
ENDCOM	198#	1988	2006	2654	3059										
ENDPAS	1705#	3554													
ERRCAL	6421#	7222	7273	7276	7551	7612									
ERREND	1705#	3622													
ERROR	92#	1914	2508	2584	2718	2732	2747	2817	2841	2905	2923	2940	3003	3021	3038
	4809	4810	4811	4812	4813	4840	4841	4842	4843	4848	4883	4884	4885	4886	4891
	4932	4933	4934	4935	4936	4962	4963	4964	4965	4966	4988	4989	4990	4991	4992
	5139	5180	5181	5182	5183	5184	5496	5501	5612	5618	5676	5677	7223	7274	7278
	7553	7613	8540												
ERRTYP	1705#	3590													
ER.NDX	1705#	4803	4834	4877	4926	4956	4982	5174							
ESCAPE	198#	2694	2795	2885	2983	4802	4833	4876	4924	4955	4981	5137	5173	5495	5499
	5611	5616	5664												
GETPRI	198#	4520	8481												
GETSMR	3#	198#	1779												
LOOP	1705#	2109	2117	2152	2156	2197	2219	2273	2317	2380	2384	2478	2516	3098	3103
	3106	3114	3121	3170	3180	3197	3203	3343	3349	3359	3421	3442	3448	3456	3462
	3470														
MORETA	468#	523													
MORE.S	1705#	2020	2054	2092	2135	2175	2255	2298	2348	2448	2605	2671	2775	2865	2963
	3077	3148	3277	3397	3486										
MSG	2011#	2013	2043#	2045	2079#	2081	2126#	2128	2165#	2167	2242#	2244	2287#	2289	2337#
	2339	2412#	2415	2592#	2594	2658#	2661	2764#	2766	2852#	2854	2950#	2952	3064#	3066
	3134#	3136	3215#	3217	3372#	3374	3477#	3479							
MULT	198#														
NEWTST	198#	2011	2043	2079	2126	2165	2242	2287	2337	2413	2592	2659	2764	2852	2950
	3064	3134	3215	3372	3477										
POP	198#	3933	4299	4488	4571	4716	4765								
PUSH	198#	3892	4279	4469	4523	4671	4729								
REPORT	198#	1705#	2755	2758	2847	2946	3044								
RPO4.D	2#	6421													
SAV. RH	1705#	2712	2726	2741	2811	2835	2899	2917	2934	2997	3015	3032			
SCOPE	93#	2041	2077	2124	2163	2240	2285	2335	2410	2590	2636	2761	2849	2948	3046
	3132	3212	3369	3473	3510										
SETPRI	198#	4123													
SETTRA	4330#	4339	4340	4341	4342	4344	4346	4347	4348	4349	4350				
SETUP	198#	1719													
SET. TN	1705#	2021	2055	2093	2136	2176	2256	2299	2349	2449	2606	2672	2776	2866	2964
	3078	3149	3278	3398	3487										
SKIP	198#														
SLASH	198#														
SPACE	198#														
STARS	68	198#	202	244	248	439	470	522	1607	2011	2019	2043	2053	2079	2091
	2126	2134	2165	2174	2242	2254	2287	2297	2337	2347	2413	2447	2592	2604	2659
	2670	2764	2774	2852	2864	2950	2962	3064	3076	3134	3147	3215	3276	3372	3396
	3477	3485	3516	3576	3626	3733	3804	3882	3950	4026	4041	4112	4136	4206	4263

	4309	4354	4373	4436	4460	4497	6420	6430	8469	8498				
SWRSU	198#	1740#												
TRNTRP	4330#													
TYPBIN	198#													
TYPB2D	1705#	5277	5284	5288	5293	5300	5304	5316	5321	6080	6155			
TYPDEC	198#	3707												
TYPEND	1705#	3523												
TYPNAM	198#													
TYPNUM	198#													
TYPOCS	198#	1807	1883	1932	3533	6049								
TYPOCT	198#	3541	3704	4054	5804	6118	6159							
TYPTXT	198#	3523	3529	3537	3554	3666	3683	5773	5853	5885	6045	6150		
SSCHRE	468#	507	508	509	510	511	512							
SSCHTH	468#	513	514	515										
SSESCA	198#													
SSNEWT	198#	2011	2043	2079	2126	2165	2242	2287	2337	2413	2592	2659	2764	2852
	3064	3134	3215	3372	3477									
SSSET	4330#	4339	4340	4341	4342	4344	4346	4347	4348	4349	4350			
SSSKIP	198#													
.EQUAT	3#	88												
.HEADE	3#													
.SETUP	3#	1709												
.SWRHI	3#	37												
.SWRLO	47#	48	49	50	51	52	53	54						
.SACT1	3#	66												
.SCATC	3#	56												
.SCMTA	3#	468												
.SDB2D	3#	4371												
.SDIV	3#	4495												
.SEOP	3#	3514												
.SERRO	3#	3574												
.SRAND	3#	4458												
.SRDEE	3#													
.SRDOC	3#													
.SREAD	3#	3948												
.SSAVE	3#	4261												
.SSB2D	3#	4352												
.SSCOP	3#	4204												
.SSIZE	3#	8467												
.SSUPR	3#	4434												
.STRAP	3#	4307												
.STYPD	3#	3880												
.STYPE	3#	3731												
.STYPO	3#	3802												

. ABS. 050554 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

NOM. SEQ/SOL/CRF/LI:ME/NL:TOC:MC:MD:CND=RPO456.010,DZRJA.014
RUN-TIME: 89 92 10 SECONDS
RUN-TIME RATIO: 710/192=3.6
CORE USED: 46K (91 PAGES)

N14

ND-11-DZRJA-AA RPO4/S/6 MECHANICAL AND READ/WRITE TEST MACY11 27(1006) 02-NOV-76 18:27 PAGE 185
DZRJA.014 19-APR-76 00:00 CROSS REFERENCE TABLE -- MACRO NAMES

