

RM02/03/05

RM05/3/2 EXT'D DR TST
CZRMVBO

AH-S074B-MC
FICHE 1 OF 2

AUG 1981
COPYRIGHT © 80-81
MADE IN USA



RM02/03/05

RM05/3/2 EXT'D DR TST
CZRMVBO

AH-S074B-MC
FICHE 2 OF 2

AUG 1981
COPYRIGHT © 80-81
MADE IN USA



.REM a

IDENTIFICATION

PRODUCT CODE: AC-S072B-MC
PRODUCT NAME: CZRMVBO RM05/3/2 EXTENDED DRIVE TEST
PRODUCT DATE: APRIL 1981
MAINTAINER: CX DIAGNOSTIC GROUP
AUTHOR: MIKE LEAVITT

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1980,1981 DIGITAL EQUIPMENT CORPORATION

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PROGRAMS
 - 2.3 MEDIA
3. LOADING PROCEDURE
4. STARTING PROCEDURE
 - 4.1 STARTING ADDRESSES
 - 4.2 OPERATOR ACTION
 - 4.3 PROGRAM ACTION
 - 4.3.1 CONTROL SWITCH SELECTION
 - 4.3.2 RH11 - RH70 ADDRESS SELECTION
 - 4.3.3 DRIVE AND PARAMETER SELECTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 CONTROL SWITCH SETTINGS
6. ERRORS
 - 6.1 ERROR TYPES
 - 6.2 ERROR RECOVERY
7. RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 TIMING TEST (TESTS 12 - 15) PRINTOUTS
 - 8.4 END OF TEST
9. PROGRAM DESCRIPTION
10. PROGRAM LISTING

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

1. ABSTRACT

THIS PROGRAM CONTAINS A SERIES OF TESTS THAT WILL VERIFY THAT THE DISK IS CAPABLE OF PERFORMING SEEKS, THAT THE ACCESS TIMES ARE WITHIN TOLERANCE AND THAT THE TRACK/SECTOR ADDRESSING CIRCUITRY OPERATES PROPERLY, AND THAT THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONING.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
12K MEMORY
TELETYPE
PROGRAM LOADING DEVICE
KW11-L OR KW11-P (THE KW11-P IS REQUIRED FOR THE TIMING TESTS)
RH11 OR RH70 CONTROLLER
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

2.2 PRELIMINARY PROGRAMS

RM05/3/2 DISKLESS TEST, PART 1 & 2
RM05/3/2 FUNCTIONAL TEST, PART 1, 2 & 3

2.3 MEDIA

THE PROGRAM REQUIRES THAT EACH DRIVE TO BE TESTED HAS A FORMATTED DISK PACK. THE PACK MAY BE FORMATTED IN EITHER 16-BIT OR 18-BIT MODE, DEPENDING ON THE TESTING REQUIREMENTS. NOTE THAT THE PROGRAM WILL NOT TEST A MIXTURE OF DRIVES WITH BOTH 16 AND 18 BIT MODE PACKS.

3. LOADING PROCEDURE

THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA USING THE ASSOCIATED LOADER.

4. STARTING PROCEDURE

4.1 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS
204 SELECT OPERATING PARAMETERS
210 SELECT RH11-RH70 ADDRESSES
214 COMBINATION OF 204 AND 210

NOTE: STARTING ADDRESSES 210 AND 214 ARE AVAILABLE WHEN THE PROGRAM IS INITIALLY STARTED; THESE STARTING ADDRESSES ARE TREATED AS ADDRESSES 200 OR 204 RESPECTIVELY ON RESTARTS.

4.2 OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3.)
2. LOAD A FORMATTED PACK INTO DRIVE(S) TO BE TESTED
3. BRING DRIVE(S) TO ONLINE STATE, WRITE ENABLED, AND LOCKED ON PORT.
4. LOAD ADDRESS 200.
5. SET SWITCHES (SEE SECTION 5.)
6. PRESS START.
7. THE PROGRAM WILL TYPEOUT THE STATUS OF THE DRIVES ATTACHED TO THE SELECTED MASSBUS SUBSYSTEM. TO INHIBIT THIS TYPEOUT, DO NOT RESTART THE PROGRAM FROM ANY OF THE STARTING ADDRESSES; INSTEAD TYPE A 'CONTROL C' ON THE KEYBOARD TO RETURN THE PROGRAM TO COMMAND ENTRY MODE.

4.3 PROGRAM ACTION

IN AN EFFORT TO ALLOW CONVERSATION WITH A PROGRAM FOR THE PURPOSE OF CONTROLLING ITS OPERATION AND PARAMETERS THE FOLLOWING CONSTRUCTIONS HAVE BEEN ADOPTED.

NOTE1. IN ALL EXAMPLES BRACKETS ARE USED FOR CLARITY AND ARE NOT TYPED BY THE USER.

NOTE2: THE CARRIAGE RETURN TYPED BY THE USER IS INDICATED BY <CR> AND WILL BE ECHOED AS A 'CARRIAGE RETURN-LINE FEED'.

<.><CR> PERIOD

A STATEMENT TERMINATOR: WHEN TYPED AT THE END OF A LINE (LEGAL ON ALL LINES) IT TELLS THE PARAMETER STRING INTERPRETER (PSI) THIS IS THE END OF CHANGES TO THE CURRENT PARAMETER STRING.

<..><CR> PERIOD PERIOD

THE 'PERIOD PERIOD' TERMINATOR IS TYPED TO INDICATE THE END OF TEST PARAMETER MODIFICATION AND TO SIGNAL THE START OF TEST EXECUTION.

<,><CR> COMMA

THE COMMA IS USED AS A SEPARATOR BETWEEN DRIVE NUMBERS AND TEST NUMBERS.

</> SLASH

A MODIFICATION INDICATOR: IF A SLASH FOLLOWS A TEST

115 NUMBER, THE PROGRAM WILL OPEN THAT TEST FOR PARAMETER
 116 MODIFICATION.
 117
 118 <^U> CONTROL-U
 119
 120 DELETE THE PRESENT INPUT STRING AND START A NEW
 121 LINE. TYPED BY DEPRESSING THE "CONTROL KEY"
 122 (CTRL) AND THEN STRIKING THE 'U'.
 123
 124 <\> RUBOUT
 125
 126 DELETE THE LAST CHARACTER FROM THE INPUT STRING.
 127 TYPED BY STRIKING THE 'RUBOUT' KEY. WHICH WILL
 128 BE ECHOED BY A BACKSLASH (\) FOLLOWED BY THE
 129 CHARACTER DELETED.
 130
 131
 132
 133
 134

4.3.1 CONTROL SWITCH SELECTION

STARTING THE PROGRAM AT ANY OF THE POSSIBLE STARTING ADDRESSES WITH SW<07>=1 WILL RESULT IN ENTERING THE 'CONTROL SWITCH SETTING' MODE. THUS, ALLOWING THE OPERATOR TO SPECIFY THE DESIRED STATE OF 'C.SWR'.

CONTROL SWITCH SELECTION EXAMPLES:

EXAMPLE #1

SET SW<07>=0
 C.SWR=000000 / 400..

EXAMPLE #2

SET SW<07>=0
 C.SWR=000000 / 220..
 C.SWR=000000 / 220..

4.3.2 RH11 - RH70 ADDRESS SELECTION

STARTING THE PROGRAM AT 200 WILL RESULT IN AUTOMATIC SELECT OF THE DEFAULT VALUES OF BUS ADDRESS (RMCS1), VECTOR ADDRESS, AND PRIORITY LEVEL OF THE RH11-RH70. IF THE DEFAULT VAULE OF THE BUS ADDRESS DOES NOT RESPOND (TIMES OUT) WHEN ADDRESSED, AN ERROR IS REPORTED. AFTER THE ERROR IS REPORTED ONE OF TWO COURSES OF ACTION WILL BE TAKEN:

1. IF THERE IS A MONITOR -- RETURN TO THE MONITOR
2. IF THERE ISN'T A MONITOR -- ASK FOR NEW ADDRESSES

STARTING THE PROGRAM AT 210 OR 214 ALLOWS THE OPERATOR TO CHANGE THE ADDRESS OF THE RH11 OR RH70 AND THE VECTOR ADDRESS.

ADDRESS SELECTION EXAMPLES

115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 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
 171

EXAMPLE #1

RMCS1=176700 / 177200.

EXAMPLE #2

RMCS1=176700 / 176300<CR>
RMVEC=254 / 260<CR>

EXAMPLE #3

RMCS1=176700<CR>
RMVEC=254 / 260.

EXAMPLE #4

RH/RM FAILED TO RESPOND TO ADDRESSING
RMCS1 ERR PC
176300 XXXXXX
RMCS1=176300 / 176700.

EXAMPLE #5

RMCS1=176700 / 1776\67\6300<CR>
RMVEC=254<CR>
RMCS1=176300.

4.3.3 DRIVE AND PARAMETER SELECTION

STARTING THE PROGRAM AT 200 OR 210 WILL RESULT IN AUTOMATIC SELECTION OF THE DRIVES TO TEST AND THE TESTS TO RUN.

STARTING THE PROGRAM AT 204 OR 214 ALLOWS THE OPERATOR TO SELECT THE DRIVE(S) TO BE TESTED, THE TESTS TO BE EXECUTED, AND THE PARAMETERS TO USE.

EACH TEST CONTAINS TWO SETS OF TRACK LIMIT PARAMETERS. PARAMETERS 'LT', 'FT' AND 'IT' ARE USED BY RM03/2 DRIVES AND PARAMETERS 'LT', 'FT' AND 'IT' ARE USED BY RM05 DRIVES. THE PROGRAM DETERMINES WHICH DRIVE IS BEING TESTED AND SELECTS THE CORRECT SET OF TRACK LIMIT VALUES. IF THE PROGRAM IS BEING USED TO TEST A SUBSYSTEM WHICH CONTAINS BOTH RM03/2 AND RM05 DRIVES, THE OPERATOR MUST CHANGE BOTH SETS OF TRACK LIMITS IF THE TESTS ARE TO BE MODIFIED FOR ALL DRIVES TESTED.

4.3.3.1 DRIVE AND PARAMETER SELECTION DESCRIPTION

THE FOLLOWING IS A TABLE OF TERMS USED BY THE PSI.

'R'	REPEATS (ITERATIONS)
'FC'	FIRST CYLINDER ADDRESS
'LC'	LAST CYLINDER ADDRESS
'IC'	INCREMENT CYLINDER
'FT'	FIRST TRACK ADDRESS
'LT'	LAST TRACK ADDRESS
'IT'	INCREMENT TRACK

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
283
284
285

'FT'' FIRST TRACK ADDRESS ;RM05 PARAMETER
'LT'' LAST TRACK ADDRESS ;RM05 PARAMETER
'IT'' INCREMENT TRACK ;RM05 PARAMETER
'FS'' FIRST SECTOR ADDRESS
'LS'' LAST SECTOR ADDRESS
'PAI'' PATTERN (USED FOR DATA TEST)
'WDX'' WORD OF PATTERN 0 WHERE X IS 1 TO 16

*'S'' ALL SEEK TESTS (TESTS 0 - 10)
*'T'' ALL TIMING TESTS (TESTS 12 - 15)
*'A'' ALL ADDRESS TESTS (TESTS 16 - 17)
*'D'' THE DATA TEST (TEST 20)
*'E'' THE EXERCISER (TEST 21)

* USED BY THE OPERATOR TO SELECT TEST GROUPS

NOTE: ALL NUMBERS WILL BE IN DECIMAL EXCEPT FOR THE PATTERN (PAT) AND WORDS (WDX) SELECTION. 'PAT' WILL BE SELECTED BY A BIT (I.E. 001000(8)=PATTERN 9) AND 'WDX' WILL BE IN OCTAL.

SPECIAL CASES OF CONTROL CHARACTERS

IF <.> IS TYPED WHILE A TEST IS OPEN FOR MODIFICATION (</>) AND OTHER TESTS IN THE 'TEST COMMAND' STRING ARE TO BE MODIFIED, THE REMAINING TESTS WILL BE UNCHANGED.

WHEN THE PROGRAM IS STARTED FROM LOCATION 200 OR 210, TESTS 0-10, 12,13,15-20 WILL BE RUN USING ALL AVAILABLE, ONLINE DRIVES. IF THE OPERATOR WISHES TO SELECT THE DRIVES TO BE TESTED, THE TESTS TO BE PERFORMED, OR THE PARAMETERS TO BE USED, THE CONVERSATION MODE MAY BE ENTERED BY TYPING A 'CONTROL C' OR BY STARTING THE PROGRAM FROM EITHER LOCATION 204 OR 214.

THE PROGRAM WILL THEN RESPOND WITH:

DRIVE(S)=

THE FOLLOWING EXAMPLES ASSUME THAT THE OPERATOR IS TO TEST DRIVE #3 USING TESTS 2 THRU 7 AND TEST 11 AND DOES NOT DESIRE TO CHANGE THE PARAMETERS (INITIAL CYLINDER ADDRESS, FINAL CYLINDER ADDRESS, ETC.). THE USER WOULD TYPE '3<CR>' WHICH SAYS 'THIS IS THE END OF DRIVE ENTRY'. THE PROGRAM WILL THEN REQUEST TEST NUMBERS.

THE TRANSACTION APPEARS AS FOLLOWS:

DRIVE(S)=3<CR>
TEST=

THE OPERATOR MAY NOW ENTER DESIRED TEST NUMBERS. IN THE EXAMPLE, HE WANTS TESTS 2 THRU 7 AND TEST 11 SO HE TYPES 2-7<,> (THE 'COMMA' SEPARATES ENTRIES), 11<.><CR> ('PERIOD' 'CARRIAGE RETURN' - END OF CHANGES, START TEST EXECUTION.)

IT NOW LOOKS LIKE THIS

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342

DRIVE(S)=3<CR>
TEST=2-7,11.<CR>

IN THE NEXT EXAMPLE, IT IS ASSUMED THAT THE OPERATOR WISHES TO TEST DRIVE 4 AND TO RUN TESTS 1 AND 3 THRU 11, MODIFYING THE PARAMETERS FOR TESTS 3 AND 10.

THE TRANSACTION WOULD BE AS FOLLOWS:

DRIVE(S)=4<CR>
TEST=

THE OPERATOR NOW ENTERS THE TEST NUMBERS. THE TRANSACTION IS GIVEN BELOW:

DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>

NOTICE THIS SAYS SELECT TEST 1, CONTINUE<, >; SELECT TEST 3, OPEN</>; SELECT TESTS 4-7, CONTINUE<, >; SELECT TEST 10, OPEN</>; SELECT TEST 11, END OF INPUT <.>.

THE PROGRAM SCANS THE TEST NUMBER INPUT AND DETERMINES THAT THE PARAMETERS FOR TEST 3 AND TEST 10 ARE TO BE CHANGED. THE OTHER TESTS WILL NOT BE ALTERED.

(THE ENTIRE TRANSACTION IS REPEATED FOR CLARITY)

DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>
TEST 3
R=X / ;WHERE X IS ITERATION

THE NEW VALUE FOR 'R' MAY BE ENTERED. TERMINATING THE ENTRY WITH A <.> (PERIOD) WILL TERMINATE THE CHANGES FOR THIS TEST; TYPING A <CR> OR TERMINATING THE ENTRY WITH A <CR> WILL CAUSE THE PROGRAM TO MOVE TO THE NEXT PARAMETER.

DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>
TEST 3
R=1 / <CR> ;DO NOT ALTER-BUT CONTINUE
FC=N / ;WHERE 'N' IS FIRST CYLINDER ADDRESS

IF THE OPERATOR DOES NOT WISH TO CHANGE 'FC', THE FOLLOWING OCCURS:

DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11.<CR>
TEST 3
R=1 / <CR> ;DO NOT ALTER THIS LINE BUT CONTINUE
FC=0 / <CR> ;DO NOT ALTER THIS LINE BUT CONTINUE
LC=822 /

THE PROGRAM RESPONDS WITH THE PREVIOUSLY ASSIGNED PARAMETER FOR LAST CYLINDER ADDRESS IN THIS CASE USING 822 AS THE EXAMPLE. THIS IS WHAT THE OPERATOR INTENDED TO MODIFY AND IS WHY TEST 3 WAS OPENED. TO CHANGE THE VALUE TO '20', THE NEW VALUE IS TYPED

FOLLOWED BY A 'PERIOD' TERMINATOR (<,><CR>).

THE TOTAL TRANSACTION AND RESPONSE:

```
DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>
TEST 3
R=1 / <CR>
FC=0 / <CR>
LC= 822 / 20.<CR>
TEST 10
R=1 /
```

THE PROGRAM HAS LOADED TEST 3 WITH ITS NEW PARAMETERS AND THE PROGRAM IS WAITING FOR CHANGES TO TEST 10'S PARAMETERS.

```
DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>
TEST 3
R=1 / <CR>
FC=0 / <CR>
LC= 822 / 20.<CR>
TEST 10
R=1 / 10.<CR>
```

THE OPERATOR TYPES THE NEW VALUE (10) AND TERMINATES THE ENTRY WITH A 'PERIOD' 'CARRIAGE RETURN'.

THE PROGRAM NOW LOADS TEST 10 WITH THE NEW PARAMETERS (TEST 11 RETAINS THE PREVIOUSLY ASSIGNED PARAMETERS) AND RESPONDS WITH:

DRIVE(S)=

SINCE THE USER DID NOT END THE CONVERSATION MODE WITH A 'PERIOD PERIOD', THE PROGRAM HAS LOOPED BACK TO THE BEGINNING LOOKING FOR MORE CHANGES. THAT IS TO SAY, AFTER THE ENTRY FOR DRIVE SELECTION, A <,><CR> WILL CAUSE THE TEST MESSAGE TO BE REPEATED AND FURTHER CHANGES CAN BE MADE. HOWEVER, AT SOME POINT IN ORDER TO EXECUTE THE PROGRAM, A 'PERIOD PERIOD' MUST BE TYPED.

IF A SINGLE 'PERIOD' IS TYPED WHILE DRIVE OR TEST NUMBERS ARE BEING ENTERED, THE PROGRAM WILL START EXECUTION IMMEDIATELY. A 'PERIOD PERIOD' MUST BE TYPED BEFORE THE PROGRAM WILL EXIT TEST PARAMETER CHANGE MODE TO GO TO EXECUTION.

4.3.3.2 DRIVE AND PARAMETER SELECTION EXAMPLES

EXAMPLE #1 -----

DRIVE=4.<CR>

:SELECT DRIVE #4. TERMINATE AND
:BEGIN EXECUTION USING PREVIOUSLY ASSIGNED
:PARAMETERS

EXAMPLE #2 -----

400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456

DRIVE=0<CR>
TEST=1-5.<CR>

;SELECT DRIVE #0 AND MAKE CHANGES ''''
;RUN TEST 1 THRU 5 ONLY, USE DEFAULT
;PARAMETERS AND TERMINATE AND EXECUTE.'

EXAMPLE #3

DRIVE=2<CR>
TEST=1-5,6/7/10/<CR>
TEST 6
R=1 / <CR>
FC=0 / 10.<CR>

;SELECT DRIVE #2 AND MAKE CHANGES ''''
;RUN TEST 1-5 WITH DEFAULT PARAMETERS, OPEN
;TEST 6,7 AND 10 FOR CHANGES
;LEAVE 'R' AS IS AND MOVE TO NEXT PARAMETER
;SET 'FC' CYLINDER ADDRESS TO 10, END CHANGES
;TO TEST 6.

TEST 7
R=1 / 50<CR>
FC=0 / <CR>
LC=822 / 50..<<CR>

;50 ITERATIONS, MOVE TO NEXT PARAMETER
;DO NOT CHANGE 'FC' CYLINDER ADDRESS BUT CONTINUE
;TEST 10 IS STILL PENDING AND WILL BE
;RETAIN ITS PRESENT PARAMETERS.

EXAMPLE #4

DRIVE=0<CR>
TEST=S,E.<CR>

;SELECT DRIVE #0 AND MAKE CHANGES
;RUN ALL SEEK TESTS AND THE EXERCISER

EXAMPLE #5

DRIVE=1<CR>
TEST=S/D<CR>
TEST 0
R=10 / <CR>
FC=0 / 10..<<CR>

;RUN ALL SEEK TESTS (OPEN FOR CHANGES) AND
;THE DATA TEST (WITH DEFAULT PARAMETERS).
;RUN WITH 10 ITERATIONS
;CHANGE FIRST CYLINDER ADDRESS
;AND START EXECUTION
;TESTS 1 - 10 WILL RETAIN THEIR PREVIOUSLY
;ASSIGNED PARAMETERS.

EXAMPLE #6

DRIVE=1<CR>
TEST=S/<CR>
TEST 0
R=10 / 100.<CR>
TEST 1
R=100 / 1000.<CR>
TEST 2
R=1 / 10<CR>
FC=0 / 50<CR>
LC=822 / 51.<CR>
TEST 3
R=1.<CR>
TEST 4
R=1..<<CR>

;OPEN THE SEEK TESTS (TESTS 0-10)
;CHANGE TO 100 ITERATIONS, GO TO THE NEXT TEST
;CHANGE 'R' TO 1000 ITERATIONS, MOVE TO NEXT TEST
;CHANGE 'R' TO 10 ITERATIONS, GO TO NEXT PARAMETER
;CHANGE 'FC' TO 50, GO TO NEXT PARAMETER
;CHANGE 'LC' TO 51, GO TO THE NEXT TEST
;MOVE TO NEXT TEST
;USE TEST 4'S PARAMETERS AND START PROGRAM EXECUTION

457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513

EXAMPLE #7

```
DRIVE=1<CR>
TEST=D/<CR>          ;SELECT AND OPEN THE DATA TEST
TEST 20
  R=1 / 1000<CR>      ;DO 1000 ITERATION OF TEST PATTERN
  FC=0 / 10<CR>      ;#8 ON CYLINDER 10, TRACK 2, SECTOR 4
  LC=822 / 10<CR>
  IC=64 / 0<CR>
  FT=0 / 2<CR>
  LT=4 / 2<CR>
  IT=1 / <CR>
  FT'=0 / 2<CR>       ;RM05 PARAMETER
  LT'=18 / 2<CR>      ;RM05 PARAMETER
  IT'=1 / <CR>        ;RM05 PARAMETER
  FS=0 / 4<CR>
  LS=22 / 4<CR>
  PAT=177777 / 400..<<CR> ;RUN WITH PATTERN #8
```

EXAMPLE #8

```
DRIVE=1<CR>          ;USE THE SAME PARAMETERS AS IN EXAMPLE
TEST=D/<CR>          ;#7, BUT ALSO SPECIFY A DATA PATTERN (PAT #0).
TEST 20
  R=1000 / <CR>
  FC=10 / <CR>
  LC=10 / <CR>
  IC=0 / <CR>
  FT=2 / <CR>
  LT=2 / <CR>
  IT=1 / <CR>
  FT'=0 / 2<CR>      ;RM05 PARAMETER
  LT'=18 / 2<CR>     ;RM05 PARAMETER
  IT'=1 / <CR>       ;RM05 PARAMETER
  FS=4 / <CR>
  LS=4 / <CR>
  PAT=000400 / 401<CR> ;RUN WITH PATTERNS #8 & #0 (O-OPERATOR INPUT)
  WD1=165555 / 125252<CR> ;FIRST WORD OF PATTERN 0
  WD2=133333 / 52525..<<CR> ;SECOND WORD OF PATTERN 0
  ;<..> START EXECUTION
```

EXAMPLE #9

```
DRIVE=0,1,4<CR>      ;TEST DRIVES 0,1, AND 4 IN SEQUENCE
TEST=0-5/<CR>        ;CHANGE TEST 5
TEST 0
  R=10 / <CR>
  FC=0 / <CR>
  LC=822 / 1..<<CR>    ;CHANGE LAST CYLINDER FROM 822 TO 1
  ;START PROGRAM EXECUTION.
```

5. SWITCH SETTINGS

5.1 OPERATIONAL SWITCH SETTINGS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176. THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS IN KEYBOARD ENTRY MODE, OR IS AT A HIGHER PRIORITY PROCESSING AN DRIVE INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTINGS ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED, IF THE PROGRAM FINDS ALL 1'S IN THE SWITCHES. ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

THE SWITCH SETTINGS ARE:

SW<15>=1	HALT ON ERROR
SW<14>=1	LOOP ON TEST
SW<13>=1	INHIBIT ERROR TYPEOUTS
SW<12>=1	TYPE TEST NUMBER
SW<11>=1	INHIBIT ITERATIONS
SW<10>=1	RING BELL ON ERROR
SW<09>=1	LOOP ON ERROR
SW<08>=1	PRINT ERROR MESSAGE ON LINE PRINTER
SW<07>=1	READ 'C.SWR' SETTINGS FROM TTY
SW<06>=1	INHIBIT TIME REPORTS (TESTS 12-15)
SW<05>=1	REPORT ONE ERROR PER SECTOR (TESTS 16 & 17)
SW<04>=1	INHIBIT WRITES (TEST 20)
SW<03>=1	INHIBIT WRITE CHECKS (TEST 20)
SW<02>=1	INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
SW<01>=1	INHIBIT SOFTWARE COMPARES (TEST 20)
SW<00>=1	PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)

5.2 CONTROL SWITCH SETTINGS

THE CONTROL SWITCH SETTINGS ARE ENTERED THROUGH THE KEYBOARD.

TO ENTER THE CONTROL SWITCH SETTING MODE PLACE SW<07>=1 BEFORE PRESSING START. THEN UPON STARTING THE PROGRAM IT WILL TYPE THE PRESENT CONTENTS OF THE CONTROL SWITCH REGISTER (C.SWR) AND WAIT FOR THE NEW SETTING TO BE INPUT. THE INPUT

571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627

STRING MUST CONSIST OF 1 TO 6 OCTAL DIGITS, TWO PERIODS (...),
AND A CARRIAGE RETURN.

THE C.SWR SETTINGS ARE:

C.SWR<15>=0	WRITE PACK BEFORE TESTING (TEST16)
=1	INHIBIT WRITE PACK BEFORE TESTING (TEST16)
C.SWR<14>=0	NO STALL BETWEEN DRIVE FUNCTIONS
=1	STALL AFTER EVERY DRIVE FUNCTION
C.SWR<13>=0	USE SPECIFIC STALL TIMES
=1	USE RANDOM STALL TIMES
C.SWR<12>=0	NO INCREMENTING STALLS IN TEST 4
=1	PERFORM INCREMENTING STALLS IN TEST 4
C.SWR<08>=0	DO IMPLIED SEEKS WITH DATA TRANSFERS
=1	DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
C.SWR<07>=0	DO READ HEADER AND DATA COMMANDS IN TESTS 0-7
=1	DO EXPLICIT SEEK COMMANDS IN TESTS 0-7
C.SWR<06>=0	60 HZ POWER SOURCE
=1	50 HZ POWER SOURCE
C.SWR<05>=0	ALLOW SOFTWARE TIMEOUTS(ENABLE WATCHDOG TIMER)
=1	INHIBIT SOFTWARE TIMEOUTS(DISABLE WATCHDOG TIMER)
C.SWR<00>=0	OPERATE IN 32. SECTOR (16 BIT) MODE
=1	OPERATE IN 30. SECTOR (18 BIT) MODE

THE DEFAULT CONDITION OF C.SWR<15:00>=0.

REFER TO 4.3.1 FOR C.SWR SELECTION

6. ERRORS

THERE ARE A NUMBER OF ERRORS THAT CAN OCCUR IN THIS PROGRAM.
WHEN AN ERROR IS ENCOUNTERED, THE CALL TO THE ERROR ROUTINE
IS MADE AND IF SW<13> IS NOT SET, AN ERROR MESSAGE PERTAINING
TO THE ERROR WILL BE TYPED. EACH ERROR TYPEOUT WILL CONTAIN
THE FOLLOWING:

1. AN ERROR MESSAGE
2. A DATA HEADER
3. A DATA STRING

REFER TO THE FOLLOWING SECTION FOR THE DIFFERENT ERRORS
THAT CAN OCCUR.

6.1 ERROR TYPES

THE ERRORS THAT OCCUR IN THIS PROGRAM FALL INTO THREE
(3) CATEGORIES DEFINED AND EXPLAINED AS FOLLOWS:

6.1.1 DRIVER ERROR

THESE ERRORS WILL BE DETECTED BY THE RH/RM DRIVER.
THERE ARE TWO CLASSES OF DRIVER ERRORS; THOSE THAT
CAN NOT BE IDENTIFIED IN A MANNER THAT ALLOWS THE

628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684

INFORMATION TO BE RETURNED TO A 'DATA PARAMETER BLOCK' (DPB) AND THOSE THAT CAN. THE FIRST CLASS WILL BE REPORTED BY ERROR CALLS (EMT'S) 1-5 WITHIN THE DRIVER. THE SECOND CLASS WILL PASS THE ERROR CODES TO THE STATUS/ERROR WORD (DPB+16) OF THE PROPER DPB.

6.1.2 NON-FATAL ERRORS

THESE ERRORS WILL BE DUE TO 'DISK' OR 'DATA' FAILURES WHICH WILL BE REPORTED AS THEY OCCUR. AFTER REPORTING THE ERROR THE PROGRAM WILL CONTINUE TESTING.

6.1.3 FATAL ERRORS

THIS TYPE OF ERROR WILL BE THE RESULT OF ANY KIND OF ERROR THAT INHIBITS THE PROGRAM FROM TESTING THE DISK.

THIS ERROR WILL BE REPORTED WHEN IT OCCURS, THEN THE PROGRAM WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

6.2 ERROR RECOVERY

6.2.1 PRETEST ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THEN DEPENDING ON HOW THE PROGRAM WAS STARTED IT WILL ASK FOR THE DRIVES AND ADDRESSES FOR TESTING OR RETURN TO MONITOR.

6.2.2 NON-FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED AND THE PROGRAM WILL CONTINUE IN TEST.

6.2.3 FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THE PROGRAM WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

7. RESTRICTIONS

THE PROGRAM WILL TEST THE DRIVES IN EITHER 16 BIT MODE OR IN 18 BIT MODE DEPENDING ON THE SETTING OF 'S.SWR<00>'. IF 'C.SWR<00>' IS 0, ALL OF THE DRIVES WILL BE TESTED IN 16 BIT MODE; IF 'C.SWR<00>' IS 1, ALL OF THE DRIVES WILL BE TESTED IN 18 BIT MODE. THE PROGRAM HAS NO PROVISIONS FOR TESTING DRIVES WITH INTERMIXED PACKS OR TESTING BOTH 16 BIT MODE AND 18 BIT MODE DRIVES ON THE SAME SYSTEM. ACT11 AUTOMATIC MODE ASSUMES 16 BIT MODE.

BEFORE THE PROGRAM IS STARTED, PROPERLY FORMATTED PACKS MUST BE MOUNTED ON THE DRIVES WHICH WILL BE TESTED. THE PROGRAM ASSUMES A PROPERLY FORMATTED PACK. THE FORMAT OF THE PACK IS NOT ALTERED BY THE PROGRAM.

685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741

8. MISCELLANEOUS

8.1 EXECUTION TIME

THE PROGRAM REQUIRES APPROXIMATELY 26 MINUTES TO MAKE ONE PASS WITH RM03/2 DRIVES AND APPROXIMATELY 35 MINUTES TO A PASS WITH RM05 DRIVES. THIS ASSUMES THE DEFAULT TEST SEQUENCE (TESTS 0-10,12,13 & 15-20) AND DEFAULT TEST PARAMETERS.

8.2 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 1100.

8.3 TIMING TESTS (TESTS 12-15) PRINTOUTS

AT THE COMPLETION OF EACH OF THE TIMING TESTS THE TIME OF THE MINIMUM SEEK, MAXIMUM SEEK, AND THE AVERAGE OF ALL OF THE SEEKS PERFORMED ARE TYPED ON THE TTY. THE NUMBER OF SEEKS THAT HAD TIMES BELOW THE MINIMUM TIME ALLOWED WILL BE TYPED ON THE SAME LINE AS THE MINIMUM TIME. THE NUMBER ABOVE THE MAXIMUM WILL BE TYPED ON THE SAME LINE AS THE MAXIMUM TIME, AND THE TOTAL NUMBER OF SEEKS PERFORMED WILL BE ON THE SAME LINE AS THE AVERAGE.

8.3.1 TIMING TOLERANCES

1. TEST 12 -- ROTATIONAL SPEED TIMES

--TIMES FOR RM05/3 DRIVES--

60 HZ
MINIMUM=16340 US
MAXIMUM=17000 US
NOMINAL=16670 US

50 HZ
MINIMUM=16250 US
MAXIMUM=17090 US
NOMINAL=16670 US

--TIMES FOR RM02 DRIVES--

60 HZ
MINIMUM=24500 US
MAXIMUM=25500 US
NOMINAL=25000 US

50 HZ
MINIMUM=24370 US
MAXIMUM=25630 US
NOMINAL=25000 US

742
 743
 744
 745
 746
 747
 748
 749
 750
 751
 752
 753
 754
 755
 756
 757
 758
 759
 760
 761
 762
 763
 764
 765
 766
 767
 768
 769
 770
 771
 772
 773
 774
 775
 776
 777
 778
 779
 780
 781
 782
 783
 784
 785
 786
 787
 788
 789
 790
 791
 792
 793
 794
 795
 796
 797
 798

2. TEST 13 -- ONE CYLINDER SEEK TIMES

MAXIMUM=6000 US

3. TEST 14 -- AVERAGE SEEK TIMING TEST (NOT DEFAULT)

MAXIMUM=30000 US

** THERE ARE NO SPECIFICATIONS GIVEN FOR AN AVERAGE SEEK TIME **
 ** ON THIS PARTICULAR DRIVE. THEREFORE, THIS TEST SHOULD BE **
 ** USED FOR REFERENCE ONLY. **

4. TEST 15 -- MAXIMUM SEEK TIMES

MAXIMUM=55000 US

** THERE IS NO SPECIFICATION GIVEN FOR THE MAXIMUM REVERSE **
 ** SEEK TIME ON THIS PARTICULAR DRIVE. THEREFORE, ANY REVERSE **
 ** SEEK TIMES ABOVE THE MAXIMUM TIME OF 55.0 MS WILL NOT BE **
 ** TYPED IN THE TIMING REPORT. HOWEVER, THE TIMING REPORT **
 ** WILL STILL TYPE THE MINIMUM, MAXIMUM AND AVERAGE TIMES **
 ** FOR THE REVERSE SEEKS. (SEE SECTION 8.3.2, EX. 2) **

8.3.2. TIMING TESTS PRINTOUT EXAMPLES

EXAMPLE #1

ROTATIONAL SPEED TIMES

MIN=16670 US

MAX=16690 US

AVG=16680 US 10 SEARCHES TIMED

ALLOWABLE ROTATIONAL SPEED LIMITS FOR RM05/3

MIN=16250 US

MAX=17090 US

ONE CYLINDER SEEK TIMES

* FORWARD

MIN=5350 US

MAX=6920 US

AVG=5550 US 821 SEEKS TIMED

* REVERSE

MIN=5140 US

MAX=5960 US

AVG=5430 US 822 SEEKS TIMED

ALLOWABLE ONE CYLINDER SEEK LIMIT

MAX=6000 US

AVERAGE SEEK TIMES

* FORWARD

MIN=27770 US

MAX=28640 US

AVG=28230 US 128 SEEKS TIMED

* REVERSE

799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855

MIN=27990 US
MAX=28550 US
AVG=28220 US 128 SEEKS TIMED

ALLOWABLE AVERAGE TIME LIMIT
MAX=30000 US

MAXIMUM SEEK TIMES

* FORWARD
MIN=49990 JS
MAX=51980 US
AVG=51010 US 128 SEEKS TIMED

* REVERSE
MIN=48120 US
MAX=50650 US
AVG=49340 US 128 SEEKS TIMED

ALLOWABLE MAXIMUM (FORWARD) SEEK TIME LIMIT
MAX=55000 US

EXAMPLE #2

ROTATIONAL SPEED TIMES

MIN=16670 US
MAX=16690 US
AVG=16680 US 10 SEARCHES TIMED

ALLOWABLE ROTATIONAL SPEED LIMITS FOR RM05/3

MIN=16250 US
MAX=17090 US

ONE CYLINDER SEEK TIMES

* FORWARD
MIN=5470 US
MAX=7940 US 3 ABOVE THE MAXIMUM OF 6000 US
AVG=5830 US 821 SEEKS TIMED
* REVERSE
MIN=5040 US
MAX=5970 US
AVG=5330 US 822 SEEKS TIMED

ALLOWABLE ONE CYLINDER SEEK LIMIT
MAX=6000 US

AVERAGE SEEK TIMES

* FORWARD
MIN=29730 US
MAX=32620 US 73 ABOVE THE MAXIMUM OF 30000 US
AVG=29900 US 128 SEEKS TIMED
* REVERSE
MIN=28620
MAX=32230 US 108 ABOVE THE MAXIMUM OF 30000 US
AVG=32800 US 128 SEEKS TIMED

ALLOWABLE AVERAGE TIME LIMIT

856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912

MAX=30000 US

MAXIMUM SEEK TIMES

* FORWARD

MIN=57510 US

MAX=57240 US 128 ABOVE THE MAXIMUM OF 55000 US

AVG=57020 US 128 SEEKS TIMED

* REVERSE

MIN=57050 US

MAX=57550 US

AVG=57210 US 128 SEEKS TIMED

ALLOWABLE MAXIMUM (FORWARD) SEEK TIME LIMIT

MAX=55000 US

8.4 END OF TEST

WITH ALL SWITCHES ON A '0' AN 'END OF PASS' MESSAGE WILL BE TYPED AT THE COMPLETION OF TESTING A DRIVE AND THE 'END OF TEST' TYPEOUT WILL OCCUR WHEN ALL DRIVES HAVE BEEN TESTED. ALSO, THE END OF TEST COULD OCCUR ON A DRIVE, IF THE MAXIMUM ERROR LIMIT IN LOCATION 'ERMAX' IS EXCEEDED.

9. PROGRAM DESCRIPTION

THIS PROGRAM CONTAINS NINETEEN TESTS NUMBERED 0-22 IN OCTAL. TESTS 0-7 & 11 WILL READ THE CYLINDER, TRACK, AND SECTOR INFORMATION FROM THE HEADER, USING A 'READ HEADER AND DATA' COMMAND, AND THEN CHECK THE INFORMATION FOR VALIDITY. THUS, INSURING THE SEEK OPERATION FUNCTIONS PROPERLY. TESTS 12-15 WILL MEASURE THE ROTATIONAL SPEED, THE ONE CYLINDER SEEK, THE AVERAGE SEEK AND THE MAXIMUM SEEK TIMES TO ENSURE THEY ARE ALL WITHIN THE TOLERANCES ALLOWED. TEST 16 AND 17 ENSURES THE SECTOR AND TRACK ADDRESSING CIRCUITRY WORKS PROPERLY. TEST 20 VERIFIES THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONAL. AND TEST 21 WILL STRESS AND CHECK THE READ/WRITE AND SERVO SYSTEMS.

THE PROGRAM WILL START BY IDENTIFYING ITSELF AND DETERMINING ALL DRIVES THAT ARE AVAILABLE FOR TESTING. THEN BEGINNING WITH THE LOWEST NUMERICAL DRIVE AND PROCEEDING IN SEQUENTIAL ORDER. ALL OF THE DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE (TESTS 0-10, 12-20) WILL BE PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE IN SEQUENCE. DRIVE TO BE TESTED WILL BE TYPED AT THE BEGINNING OF EACH PASS, AN 'END OF PASS' MESSAGE WILL BE TYPED AT THE COMPLETION OF EACH PASS, AND AN 'END OF TEST' MESSAGE WILL BE TYPED AFTER TESTING ALL DRIVES.

REFER TO THE FOLLOWING SECTIONS FOR DETAILED DESCRIPTIONS OF EACH TEST.

9.1 TEST 0 - RECAL/RANDOM SEEK TEST

913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969

THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE COMMAND AND THEN SEEK TO A RANDOM CYLINDER BETWEEN 'FC' AND 'LC'. AT THE COMPLETION OF BOTH COMMANDS, STATUS INDICATORS ARE CHECKED TO ENSURE THAT NO ERRORS OCCURRED.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	200
FC	-	0
LC	-	822
FT	-	0
FT'	-	0
FS	-	0

9.2 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.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	100
FC	-	0
LC	-	256
IC	-	0
FT	-	0
LT	-	0
FT'	-	0
LT'	-	0
FS	-	0
LS	-	0

9.3 TEST 2 - INCREMENTAL 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.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.4 TEST 3 - STEPPING SEEK TEST

970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026

THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4, 8,16,32,64,128,256 AND 512. AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO VERIFY PROPER OPERATION.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	8
FC	-	0
LC	-	512
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.5 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.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.6 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.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.7 TEST 6 - SERVO ADDRESSING LOGIC NOISE GENERATOR TEST

IN THIS TEST A SEEK IS DONE TO CYL 'NC' THEN A SEEK 'O

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 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.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.8 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'.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	5000
FC	-	0
LC	-	822
FT	-	0
LT	-	4
FT'	-	0
LT'	-	18

9.9 TEST 1C - 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 (RMLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING INTO POSITION. THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND FOR THAT SECTOR. 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 IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS. THE NECESSARY TIME DEPENDENT PARAMETERS OCCUR WITHIN THE REQUIRED TIME RANGE FREQUENTLY ENOUGH TO PERMIT THIS TEST TO BE EFFECTIVE.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083

1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140

R	-	1
FC	-	0
LC	-	822
IC	-	100
FT	-	0
FT'	-	0

9.10 TEST 11 - ALL SEEKS TEST (NOT DEFAULT)

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.

THE FOLLOWING PARAMETERS ARE USED BY THIS TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.11 TEST 12 - ROTATIONAL SPEED TIMING TEST

THIS TEST WILL START A SEARCH TO CYLINDER 'FC', TRACK 'FT', SECTOR 'FS'. 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:

RM05/3:

16.67 MS/REV + OR - 2% IF 60HZ
16.67 MS/REV + OR - 2.5% IF 50HZ.

RM02:

25.00 MS/REV + OR - 2% IF 60HZ
25.00 MS/REV + OR - 2.5% IF 50HZ.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
FT	-	0
FT'	-	0
FS	-	0

9.12 TEST 13 - ONE CYLINDER SEEK TIMING TEST

THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE

CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'. THE TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK. MAXIMUM TIME IS 6.0 MS.

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
LC	-	822

9.13 TEST 14 - AVERAGE SEEK TIMING TEST (NOT DEFAULT)

THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO CYLINDER 'LC', THEN A REVERSE FROM CYLINDER 'LC' TO CYLINDER 'FC'. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE AVERAGE SEEK. THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF 256. SEEKS). MAXIMUM TIME IS 30.0 MS.

** THERE ARE NO SPECIFICATIONS GIVEN FOR AN AVERAGE SEEK TIME **
** ON THIS PARTICULAR DRIVE. THEREFORE, THIS TEST SHOULD BE **
** USED FOR REFERENCE ONLY. **

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
LC	-	220

9.14 TEST 15 - MAXIMUM SEEK TIMING TEST

THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO CYLINDER 'FC'. BOTH SEEKS ARE TIMED, BUT ONLY THE FORWARD SEEKS ARE CHECKED TO ENSURE THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK TIME. THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF 256. SEEKS). THE MAXIMUM (FORWARD) TIME IS 55.0 MS.

** THERE IS NO SPECIFICATION GIVEN FOR THE MAXIMUM REVERSE **
** SEEK TIME ON THIS PARTICULAR DRIVE. THEREFORE, ANY REVERSE **
** SEEK TIMES ABOVE THE MAXIMUM TIME OF 55.0 MS WILL NOT BE **
** TYPED IN THE TIMING REPORT. HOWEVER, THE TIMING REPORT **
** WILL STILL TYPE THE MINIMUM, MAXIMUM AND AVERAGE TIMES **
** FOR THE REVERSE SEEKS. (SEE SECTION 8.3.2, EX. 2) **

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
LC	-	822

9.15 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

1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254

CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED. THEN SECTOR IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED. THEN SECTOR 1 IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED. THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH REWRITE SECTOR 31 AND WRITE CHECK SECTORS 0-31.

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
FT	-	0
FT'	-	0

9.16 TEST 17 - TRACK ADDRESSING TEST

THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES IN CYLINDER 'FC' SECTOR 'FS' OF EVERY TRACK WITH EACH TRACK GETTING ITS OWN TRACK ADDRESS. A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO INSURE THE DATA IS VALID. THEN TRACK 0 IS REWRITTEN AND TRACK 1 THROUGH LAST TRACK IS WRITE CHECKED. THEN TRACK 1 IS REWRITTEN AND TRACK 2 THROUGH LAST TRACK IS WRITE CHECKED. THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING NEXT TO LAST TRACK AND WRITE CHECKING LAST TRACK.

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
FS	-	0

9.17 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 177777 (ALL POSSIBLE PATTERNS)
THE POSSIBLE PATTERNS ARE:

*PAT 0	PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7
155555	000001	177776	000000	133331	052525	155555	026455
133333	000003	177774	000000	133331	052525	155555	026455
155555	000007	177770	000000	133331	052525	155555	026455

1255	133333	000017	177760	177777	133331	125252	155555	151322
1256	155555	000037	177740	177777	133331	125252	155555	151322
1257	133333	000077	177700	177777	133331	125252	155555	151322
1258	155555	000177	177600	000000	133331	052525	155555	026455
1259	133333	000377	177400	000000	133331	052525	155555	026455
1260	155555	000777	177000	177777	133331	125252	155555	151322
1261	133333	001777	176000	177777	133331	125252	155555	151322
1262	155555	003777	174000	000000	133331	052525	155555	026455
1263	133333	007777	170000	177777	133331	125252	155555	151322
1264	155555	017777	160000	000000	133331	052525	155555	026455
1265	133333	037777	140000	177777	133331	125252	155555	151322
1266	155555	077777	100000	000000	133331	052525	155555	026455
1267	133333	177777	000000	177777	133331	125252	155555	151322
1268								
1269	*PAT 8	PAT 9	PAT 10	PAT 11	PAT 12	PAT 13	PAT 14	PAT 15
1270	-----	-----	-----	-----	-----	-----	-----	-----
1271	155555	000001	177776	172666	077777	153333	000000	177777
1272	133333	000002	177775	155555	137777	066667	177777	000000
1273	155555	000004	177773	172666	157777	153333	177777	000000
1274	133333	000010	177767	155555	167777	066667	177777	000000
1275	155555	000020	177757	172666	173777	153333	177777	000000
1276	133333	000040	177737	155555	175777	066667	177777	000000
1277	155555	000100	177677	172666	176777	153333	177777	000000
1278	133333	000200	177577	155555	177377	066667	177777	000000
1279	155555	000400	177377	172666	177577	153333	177777	000000
1280	133333	001000	176777	155555	177677	066667	177777	000000
1281	155555	002000	175777	172666	177737	153333	177777	000000
1282	133333	004000	173777	155555	177757	066667	177777	000000
1283	155555	010000	167777	172666	177767	153333	177777	000000
1284	133333	020000	157777	155555	177773	066667	177777	000000
1285	155555	040000	137777	172666	177775	153333	177777	000000
1286	133333	100000	077777	155555	177776	066667	177777	000000

* WORST CASE PATTERN

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
LC	-	822
IC	-	64
FT	-	0
LT	-	4
IT	-	1
FT'	-	0
LT'	-	18
IT'	-	1
FS	-	1
LS	-	0
PAT	-	177777

9.18 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.

1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311

1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352

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

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	20000
FC	-	0
LC	-	822

9.19 TEST 22 - SEEK TIME ADJUSTMENT TEST

THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 255 TO ALLOW THE OPERATOR TO ADJUST THE SEEK TIME ON AN RM05/3/2 USING THE DDU. THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS SO THAT THE SEEK TIME INDICATORS ON THE DDU MAY BE OBSERVED.

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	5000
FC	-	0
LC	-	255

10. PROGRAM LISTING

a

LAST REVISION 04-APR-81

TITLE CZPMVBO RM05/3/2 EXT'D DR TST

*COPYRIGHT (C) 1981
*DIGITAL EQUIPMENT CORPORATION
*COLORADO SPGS., CO. 80919

*PROGRAM BY MIKE LEAVITT

*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81

SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
12	TYPE TEST NUMBER
10	BELL ON ERROR
9	LOOP ON ERROR
8	PRINT ERROR MESSAGE ON LINE PRINTER
7	READ 'C.SWR' SETTINGS FROM TTY
6	INHIBIT TIME REPORTS (TESTS 12-15)
5	REPORT ONE ERROR PER SECTOR (TESTS 16 & 17)
4	INHIBIT WRITES (TEST 20)
3	INHIBIT WRITE CHECKS (TEST 20)
2	INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
1	INHIBIT SOFTWARE COMPARES (TEST 20)
0	PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)

SBTTL CONTROL SWITCH SETTINGS

SWITCH	STATE	USE
15	0	WRITE PACK BEFORE TESTING (TEST 21)
	1	INHIBIT WRITING PACK BEFORE TESTING (TEST 21)
14	0	NO STALL BETWEEN DRIVE FUNCTIONS
	1	STALL AFTER EVERY DRIVE FUNCTION
13	0	USE SPECIFIC STALL TIME
	1	USE RANDOM STALL
12	0	NO INCREMENTING STALL IN TEST 4
	1	DO INCREMENTING STALL IN TEST 4
8	0	DO IMPLIED SEEKS WITH DATA TRANSFERS
	1	DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
7	0	DO 'READ HEADER AND DATA' IN TESTS 0-11
	1	DO EXPLICIT SEEKS IN TESTS 0-11
6	0	60 HZ
	1	50 HZ
5	0	RUN WATCHDOG TIMER
	1	INHIBIT WATCHDOG TIMER
0	0	TEST DRIVE(S) IN 32. SECTOR (16 BIT) MODE
	1	TEST DRIVE(S) IN 30. SECTOR (8 BIT) MODE

SBTTL BASIC DEFINITIONS

```

; *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100  STACK = 1100
104000  ERROR = EMT      ;;BASIC DEFINITION OF ERROR CALL
000004  SCOPE = IOT      ;;BASIC DEFINITION OF SCOPE CALL

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

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

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

; *'SWITCH REGISTER' SWITCH DEFINITIONS
100000  SW15 = 100000
040000  SW14 = 40000
020000  SW13 = 20000
010000  SW12 = 10000
004000  SW11 = 4000
002000  SW10 = 2000
001000  SW09 = 1000
000400  SW08 = 400
000200  SW07 = 200
000100  SW06 = 100
000040  SW05 = 40
000020  SW04 = 20
000010  SW03 = 10
000004  SW02 = 4
000002  SW01 = 2
000001  SW00 = 1
001000  SW9-SW09

```

000400 SW8=SW08
 000200 SW7=SW07
 000100 SW6=SW06
 000040 SW5=SW05
 000020 SW4=SW04
 000010 SW3=SW03
 000004 SW2=SW02
 000002 SW1=SW01
 000001 SW0=SW00

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

100000 BIT15 = 100000
 040000 BIT14 = 40000
 020000 BIT13 = 20000
 010000 BIT12 = 10000
 004000 BIT11 = 4000
 002000 BIT10 = 2000
 001000 BIT09 = 1000
 000400 BIT08 = 400
 000200 BIT07 = 200
 000100 BIT06 = 100
 000040 BIT05 = 40
 000020 BIT04 = 20
 000010 BIT03 = 10
 000004 BIT02 = 4
 000002 BIT01 = 2
 000001 BIT00 = 1
 001000 BIT9=BIT09
 000400 BIT8=BIT08
 000200 BIT7=BIT07
 000100 BIT6=BIT06
 000040 BIT5=BIT05
 000020 BIT4=BIT04
 000010 BIT3=BIT03
 000004 BIT2=BIT02
 000002 BIT1=BIT01
 000001 BIT0=BIT00

;*BASIC "CPU" TRAP VECTOR ADDRESSES

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

.SBTTL RH REGISTERS

;CONTROL AND STATUS REGISTER 1 (RMCS1)

000100 IE = 100 ;:INTERRUPT ENABLE (BIT #6)

72
 73
 74
 75
 76
 77

78	000200	RDY	= 200	;READY (BIT #7)
79	000400	A16	= 400	;HIGH ORDER BUS ADDRESS BIT (BIT #8)
80	001000	A17	= 1000	;HIGH ORDER BUS ADDRESS BIT (BIT #9)
81	002000	PSEL	= 2000	;PORT SELECT (BIT #10)
82	020000	MCPE	= 20000	;MASSBUS PARITY ERROR (BIT #13)
83	040000	TRE	= 40000	;TRANSFER ERROR (BIT #14)
84		;SC	= 100000	;SPECIAL CONDITION (BIT #15)
85				
86		;WORD COUNT REGISTER (RMWC)		
87		;(EACH BIT IS CALLED BY BIT NUMBER)		
88				
89		;BUS ADDRESS REGISTER (RMBA)		
90		;(EACH BIT IS CALLED BY BIT NUMBER)		
91				
92		;CONTROL AND STATUS REGISTER 2 (RMCS2)		
93				
94	000001	US1	= 1	;UNIT SELECT (BIT #0)
95	000002	US2	= 2	;UNIT SELECT (BIT #1)
96	000004	US4	= 4	;UNIT SELECT (BIT #2)
97	000010	BAI	= 10	;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
98		;PAT	= 20	;MASSBUS PARITY TEST (BIT #4)
99	000040	CLR	= 40	;CLEAR (BIT #5)
100	000100	IR	= 100	;INPUT READY (BIT #6)
101	000200	OR	= 200	;OUTPUT READY (BIT #7)
102	000400	MPE	= 400	;MASS BUS PARITY ERROR (BIT #8)
103	001000	MXF	= 1000	;MISSED TRANSFER ERROR (BIT #9)
104	002000	PGE	= 2000	;PROGRAM ERROR (BIT #10)
105	004000	NEM	= 4000	;NON EXISTENT MEMORY (BIT #11)
106	010000	NED	= 10000	;NON EXISTENT DRIVE (BIT #12)
107	020000	UPE	= 20000	;UNIBUS PARITY ERROR (BIT #13)
108	040000	WCE	= 40000	;WRITE CHECK ERROR (BIT #14)
109	100000	DLT	= 100000	;DATA LATE (BIT #15)
110				
111		;DATA BUFFER REGISTER (RMDB)		
112		;(EACH BIT IS CALLED BY BIT NUMBER)		
113				
114		.SBTTL RM REGISTERS		
115				
116		;CONTROL AND STATUS 1 REGISTER. (#00)		
117				
118	000001	G0	= 1	;GO BIT (BIT #0)
119	000002	F1	= 2	;FUNCTION CODE BIT #1
120	000004	F2	= 4	;FUNCTION CODE BIT #2
121	000010	F3	= 10	;FUNCTION CODE BIT #3
122	000020	F4	= 20	;FUNCTION CODE BIT #4
123	000040	F5	= 40	;FUNCTION CODE BIT #5
124	004000	DVA	= 4000	;DEVICE AVAILABLE (BIT #11)
125				
126		;DRIVE STATUS REGISTER (RMDS, (#01)		
127				
128	000001	OM	= 1	;OFFSET MODE
129	000100	VV	= 100	;VOLUME VALID (BIT #6)
130	000200	DRY	= 200	;DRIVE READY (BIT #7)
131	000400	DPR	= 400	;DRIVE PRESENT (BIT #8)
132	001000	PGM	= 1000	;PROGRAMABLE (BIT #9)
133	002000	LST	= 2000	;LAST SECTOR TRANSFERRED (BIT #10)
134	004000	WRL	= 4000	;WRITE LOCK (BIT #11)

RM REGISTERS

135	010000	MOL	= 10000	;MEDIUM ON-LINE (BIT #12)
136	020000	PIP	= 20000	;POSITIONING OPERATION IN PROGRESS (BIT #13)
137	040000	ERR	= 40000	;COMPOSITE ERROR (BIT #14)
138	100000	ATA	= 100000	;ATTENTION ACTIVE (BIT #15)
139				
140		;ERROR REGISTER #01 (RMER1) (#02)		
141				
142	000001	ILF	= 1	;ILLEGAL FUNCTION (BIT #0)
143	000002	ILR	= 2	;ILLEGAL REGISTER (BIT #1)
144	000004	RMR	= 4	;REGISTER MODIFICATION REFUSED (BIT #2)
145	000010	PAR	= 10	;PARITY ERROR (BIT #3)
146	000020	FER	= 20	;FORMAT ERROR (BIT #4)
147	000040	WCF	= 40	;WRITE CLOCK FAIL (BIT #5)
148	000100	ECH	= 100	;ECC HARD ERROR (BIT #6)
149	000200	HCE	= 200	;HEADER COMPARE ERROR (BIT #7)
150	000400	HCRC	= 400	;HEADER CRC ERROR (BIT #8)
151	001000	AOE	= 1000	;ADDRESS OVERFLOW ERROR (BIT #9)
152	002000	IAE	= 2000	;INVALID ADDRESS ERROR (BIT #10)
153	004000	WLE	= 4000	;WRITE LOCK ERROR (BIT #11)
154	010000	DTE	= 10000	;DRIVE TIMING ERROR (BIT #12)
155	020000	OP!	= 20000	;OPERATION INCOMPLETE (BIT #13)
156	040000	UNS	= 40000	;DRIVE UNSAFE (BIT #14)
157	100000	DCK	= 100000	;DATA CHECK ERROR (BIT 15)
158				
159		;MAINTAINABILITY REGISTER #01 (RMMR1) (#03) - READ ONLY BITS		
160				
161	000001	DMD	= 1	;DIAGNOSTIC MODE
162	000002	LSIT	= 2	
163		;LS	= 4	
164	000010	WD	= 10	
165	000020	EECC	= 20	
166	000040	WC	= 40	
167	000100	CONT	= 100	
168	000200	PHA	= 200	
169	000400	PDA	= 400	
170	001000	ECRC	= 1000	
171	002000	PLFS	= 2000	
172	004000	ESRC	= 4000	
173	010000	REX	= 10000	
174	020000	EBL	= 20000	
175		;R/G	= 40000	
176	100000	OCC	= 100000	
177				
178		;MAINTAINABILITY REGISTER #01 (RMMR1) (#03) - WRITE ONLY BITS		
179				
180	000001	DMD	= 1	;DIAGNOSTIC MODE BIT
181	000002	MSC	= 2	
182	000004	MJ	= 4	
183	000010	MWP	= 10	
184	000020	DTG	= 20	
185	000040	MS	= 40	
186	000100	MDF	= 100	
187	000200	MSER	= 200	
188	000400	MOC	= 400	
189	001000	MUR	= 1000	
190	002000	MRD	= 2000	
191	004000	MCLK	= 4000	

192	010000	MSEN	= 10000	
193	020000	DT0	= 20000	
194	040000	OBSEN	= 40000	
195	100000	OBCK	= 100000	
196				
197		;ATTENTION SUMMARY PSEUDO-REGISTER (RMAS) (#04)		
198				
199	000001	AT0	= 1	;DEVICE 0 (BIT #0)
200	000002	AT1	= 2	;DEVICE 1 (BIT #1)
201	000004	AT2	= 4	;DEVICE 2 (BIT #2)
202	000010	AT3	= 10	;DEVICE 3 (BIT #3)
203	000020	AT4	= 20	;DEVICE 4 (BIT #4)
204	000040	AT5	= 40	;DEVICE 5 (BIT #5)
205	000100	AT6	= 100	;DEVICE 6 (BIT #6)
206	000200	AT7	= 200	;DEVICE 7 (BIT #7)
207				
208		;DESIRED SECTOR/TRACK ADDRESS REGISTER (RMDA) (#05)		
209		;(EACH BIT IS CALLED BY BIT NUMBER)		
210				
211		;DRIVE TYPE REGISTER (RMDT) (#06)		
212				
213	000001	DT00	= 1	;DRIVE TYPE NUMBER BIT 1
214	000002	DT01	= 2	;DRIVE TYPE NUMBER BIT 2
215	000004	DT02	= 4	;DRIVE TYPE NUMBER BIT 3
216	000010	DT03	= 10	;DRIVE TYPE NUMBER BIT 4
217	000020	DT04	= 20	;DRIVE TYPE NUMBER BIT 5
218	000040	DT05	= 40	;DRIVE TYPE NUMBER BIT 6
219	000100	DT06	= 100	;DRIVE TYPE NUMBER BIT 7
220	000200	DT07	= 200	;DRIVE TYPE NUMBER BIT 8
221	000400	DT08	= 400	;DRIVE TYPE NUMBER BIT 9
222	004000	DRQ	= 4000	;DRIVE REQUEST REQUIRED (BIT #11)
223	020000	MOH	= 20000	;MOVING HEAD (BIT #13)
224	040000	TAP	= 40000	;TAPE DRIVE (BIT #14)
225	100000	NBA	= 100000	;NOT BLOCK ADDRESSED (BIT #15)
226				
227		;LOOK-AHEAD REGISTER (RMLA) (#07)		
228				
229				
230	000100	SC0	= 100	;SECTOR COUNT FIELD 0 (BIT #6)
231	000200	SC1	= 200	;SECTOR COUNT FIELD 1 (BIT #7)
232	000400	SC2	= 400	;SECTOR COUNT FIELD 2 (BIT #8)
233		;SC3	= 1000	;SECTOR COUNT FIELD 3 (BIT #9)
234		;SC4	= 2000	;SECTOR COUNT FIELD 4 (BIT #10)
235				
236				
237		;RM MAINTAINABILITY REGISTER #2 (RMMR2) (#10)		
238				
239		;RM ERROR REGISTER #02 (RMMR2) (#10)		
240				
241		;OFFSET REGISTER (RMOF) (#11)		
242				
243	000200	OFD	= 200	;OFFSET DIRECTION (BIT #07)
244	002000	HCI	= 2000	;HEADER COMPARE INHIBIT (BIT #10)
245	004000	ECI	= 4000	;ERROR CORRECTION CODE INHIBIT (BIT #11)
246	010000	FMT16	= 10000	;FORMAT BIT (BIT #12)
247				
248		;DESIRED CYLINDER ADDRESS (RMDC) (#12)		

```

249      : (EACH BIT IS CALLED BY BIT NUMBER)
250
251      : CURRENT CYLINDER ADDRESS (RMHR) (#13)
252      : (EACH BIT IS CALLED BY BIT NUMBER)
253
254      : SERIAL NUMBER REGISTER (RMSN) (#14)
255      : (EACH IS CALLED BY BIT NUMBER)
256
257
258      : RM ERROR REGISTER #02 (RMER2) (#15)
259
260      020000      OPE      = 20000      : OPERATOR PLUG ERROR (BIT #13)
261      040000      SKI      = 40000      : SEEK INCOMPLETE (BIT #14)
262      100000      BSE      = 100000     : BAD SECTOR ERROR (BIT #15)
263
264      : ECC POSITION REGISTER (RMEC1) (#16)
265      : (EACH BIT IS CALLED BY BIT NUMBER)
266
267      : ECC PATTERN REGISTER (RMEC2) (#17)
268      : (EACH BIT IS CALLED BY BIT NUMBER)
269
270      : *****
271
272      : OP CODE DEFINITIONS
273      000101      NOOP      = 101
274      000103      UNLOAD    = 103
275      000105      SEEK      = 105
276      000107      RECAL     = 107
277      000111      DRVCLR    = 111
278      000113      RELEASE   = 113
279      000115      OFFSET    = 115
280      000117      RTC       = 117
281      000121      READIN    = 121
282      000123      PACK      = 123
283      000131      SEARCH    = 131
284      000151      WRCKD     = 151
285      000153      WRCKHD    = 153
286      000161      WRITE     = 161
287      000163      WRTHD     = 163
288      000171      READ      = 171
289      000173      READHD    = 173
290      000141      GETREG    = 141
291      000143      SETFORM   = 143
292      000145      SELDRV    = 145
293
294      : OTHER EQUATES
295
296      177400      SCTRWC     = -256.      : WORD COUNT FOR SECTOR
297
  
```

```

      .SBTTL TRAP CATCHER

      000000      .=0
                  ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                  ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                  ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

      000174      000174
      000176      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
      000176      000000      SWREG: .WORD 0      ;;SOFTWARE SWITCH REGISTER

      .SBTTL STARTING ADDRESS(ES)

      000200      000137      004670      JMP      @#START1      ;;JUMP TO STARTING ADDRESS OF PROGRAM
2
3      000204      000137      004712      JMP      @#START2      ;SELECT OPERATING PARAMETERS
4
5      000210      000137      004660      JMP      @#START3      ;SELECT RH/RM ADDRESSES
6
7      000214      000137      004702      JMP      @#START4      ;COMBINATION OF 204 AND 210
8
9
      .SBTTL ACT11 HOOKS

      ;*****
      ;HOOKS REQUIRED BY ACT11
      000220      000046      $SVPC=.      ;SAVE PC
      000046      021560      .=46      ;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
      000052      020000      $ENDAD      ;2)SET LOC.52 TO 20000
      000220      .WORD 20000      ;: RESTORE PC
      000220      .=$SVPC

10
11      . 1100
12      .SBTTL APT PARAMETER BLOCK

      ;*****
      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
      ;*****
      000024      001100      .SX=.      ;;SAVE CURRENT LOCATION
      000024      000024      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
      000024      000200      200      ;;FOR APT START UP
      000044      000044      .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
      000044      001100      $APTHDR ;;POINT TO APT HEADER BLOCK
      001100      001100      .=$X      ;;RESET LOCATION COUNTER

      ;*****
      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
      ;INTERFACE SPEC.

      001100      $APTHD:
      001100      000000      $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
      001102      001234      $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
      001104      004064      $TSTM: .WORD 2100. ;;RUN TIM OF LONGEST TEST
      001106      004064      $PASTM: .WORD 2100. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
      001110      004064      $UNITM: .WORD 2100. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
      001112      000030      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
13
14      TAB.XY=.      ;COMMON TAG STARTING ADDRESS
  
```

0

.SBTTL COMMON TAGS

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

001114	001114	SCMTAG:	.=TAB.XY	::START OF COMMON TAGS
001114	000000		.WORD 0	
001116	000	\$TSTNM:	.BYTE 0	::CONTAINS THE TEST NUMBER
001117	000	\$ERFLG:	.BYTE 0	::CONTAINS ERROR FLAG
001120	000000	\$ICNT:	.WORD 0	::CONTAINS SUBTEST ITERATION COUNT
001122	000000	\$LPADR:	.WORD 0	::CONTAINS SCOPE LOOP ADDRESS
001124	000000	\$LPERR:	.WORD 0	::CONTAINS SCOPE RETURN FOR ERRORS
001126	000000	\$ERTTL:	.WORD 0	::CONTAINS TOTAL ERRORS DETECTED
001130	000	\$ITEMB:	.BYTE 0	::CONTAINS ITEM CONTROL BYTE
001131	001	\$ERMAX:	.BYTE 1	::CONTAINS MAX. ERRORS PER TEST
001132	000000	\$ERRPC:	.WORD 0	::CONTAINS PC OF LAST ERROR INSTRUCTION
001134	000000	\$GDADR:	.WORD 0	::CONTAINS ADDRESS OF 'GOOD' DATA
001136	000000	\$BDADR:	.WORD 0	::CONTAINS ADDRESS OF 'BAD' DATA
001140	000000	\$GDDAT:	.WORD 0	::CONTAINS 'GOOD' DATA
001142	000000	\$BDDAT:	.WORD 0	::CONTAINS 'BAD' DATA
001144	000000		.WORD 0	::RESERVED--NOT TO BE USED
001146	000000		.WORD 0	
001150	000	\$AUTOB:	.BYTE 0	::AUTOMATIC MODE INDICATOR
001151	000	\$INTAG:	.BYTE 0	::INTERRUPT MODE INDICATOR
001152	000000		.WORD 0	
001154	177570	\$WR:	.WORD DSWR	::ADDRESS OF SWITCH REGISTER
001156	177570	\$DISPLAY:	.WORD DDISP	::ADDRESS OF DISPLAY REGISTER
001160	177560	\$TKS:	177560	::TTY KBD STATUS
001162	177562	\$TKB:	177562	::TTY KBD BUFFER
001164	177564	\$TPS:	177564	::TTY PRINTER STATUS REG. ADDRESS
001166	177566	\$TPB:	177566	::TTY PRINTER BUFFER REG. ADDRESS
001170	000	\$NULL:	.BYTE 0	::CONTAINS NULL CHARACTER FOR FILLS
001171	002	\$FILLS:	.BYTE 2	::CONTAINS # OF FILLER CHARACTERS REQUIRED
001172	012	\$FILLC:	.BYTE 12	::INSERT FILL CHARS. AFTER A 'LINE FEED'
001173	000	\$TPFLG:	.BYTE 0	::'TERMINAL AVAILABLE' FLAG (BIT<07>-0 YES)
001174	000000	\$REGAD:	.WORD 0	::CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
001176	000000	\$REG0:	.WORD 0	::CONTAINS ((\$REGAD)+0)
001200	000000	\$REG1:	.WORD 0	::CONTAINS ((\$REGAD)+2)
001202	000000	\$REG2:	.WORD 0	::CONTAINS ((\$REGAD)+4)
001204	000000	\$REG3:	.WORD 0	::CONTAINS ((\$REGAD)+6)
001206	000000	\$REG4:	.WORD 0	::CONTAINS ((\$REGAD)+10)
001210	000000	\$REG5:	.WORD 0	::CONTAINS ((\$REGAD)+12)
001212	000000	\$TMP0:	.WORD 0	::USER DEFINED
001214	000000	\$TMP1:	.WORD 0	::USER DEFINED
001216	000000	\$TMP2:	.WORD 0	::USER DEFINED
001220	000000	\$TIMES:	0	::MAX. NUMBER OF ITERATIONS
001222	000000	\$ESCAPE:	0	::ESCAPE ON ERROR ADDRESS
001224	207	\$BELL:	.ASCII <207><377><377>	::CODE FOR BELL
001230	077	\$QUES:	.ASCII /?/	::QUESTION MARK
001231	015	\$CRLF:	.ASCII <15>	::CARRIAGE RETURN
001232	012	\$LF:	.ASCII <12>	::LINE FEED

 .SBTTL APT MAILBOX-ETABLE

```

*****
.EVEN
001234 $MAIL:
001234 000000 $MSGTY: .WORD AMSGTY ;;APT MAILBOX
001236 000000 $FATAL: .WORD AFATAL ;;MESSAGE TYPE CODE
001240 000000 $TESTN: .WORD ATESTN ;;FATAL ERROR NUMBER
001242 000000 $PASS: .WORD APASS ;;TEST NUMBER
001244 000000 $DEVCT: .WORD ADEVCT ;;PASS COUNT
001246 000000 $UNIT: .WORD AUNIT ;;DEVICE COUNT
001250 000000 $MSGAD: .WORD AMSGAD ;;I/O UNIT NUMBER
001252 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE ADDRESS
001254 $ETABLE: ;;MESSAGE LENGTH
001254 000 $ENV: .BYTE AENV ;;APT ENVIRONMENT TABLE
001255 000 $ENVN: .BYTE AENVN ;;ENVIRONMENT BYTE
001256 000000 $SWREG: .WORD ASWREG ;;ENVIRONMENT MODE BITS
001260 000000 $USWR: .WORD AUSWR ;;APT SWITCH REGISTER
001262 000000 $CPUOP: .WORD ACPUOP ;;USER SWITCHES
;;CPU TYPE,OPTIONS
BITS 15-11=CPU TYPE
11/04=01,11/05=02,11/20=03,11/40 04,11/45 05
11/70=06,PDQ=07,Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT
001264 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
001265 000 $MTYP1: .BYTE AMTYP1 ;;MEM. TYPE,BLK#1
;;MEM. TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003
001266 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
;;MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
001270 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
001271 000 $MTYP2: .BYTE AMTYP2 ;;MEM. TYPE,BLK#2
001272 000000 $MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
001274 000 $MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
001275 000 $MTYP3: .BYTE AMTYP3 ;;MEM. TYPE,BLK#3
001276 000000 $MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
001300 000 $MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS M.S.BYTE
001301 000 $MTYP4: .BYTE AMTYP4 ;;MEM. TYPE,BLK#4
001302 000000 $MADR4: .WORD AMADR4 ;;MEM.LAST ADDRESS,BLK#4
001304 000000 $VECT1: .WORD AVECT1 ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001306 000000 $VECT2: .WORD AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001310 000000 $BASE: .WORD ABASE ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001312 000000 $DEVN: .WORD ADEVN ;;DEVICE MAP
001314 $ETEND:
.MEXIT

```

.SBTTL USER DEFINED TAGS

001314	000000	C.SWR: .WORD	0	:CONTROL SWITCHES
001316	000031	ERMAX: .WORD	25.	:MAXIMUM NUMBER OF ERRORS ALLOWED PER DRIVE
001320	000000	SAVCSW: .WORD	0	:PREVIOUS CONTENTS OF 'C.SWR'
001322	000000	CNTRLC: .WORD	0	:CONTROL 'C' FLAG
001324	000000	BUSADR: .WORD	0	:GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
001326	000000	LPTAVL: .WORD	0	:LPT AVAILABLE STATUS (0=NO, 1=YES)
001330	000000	DRVSEL: .WORD	0	:DRIVES SELECTED FOR TESTING
001332	166777	TSTNMS: .WORD	166777,3	:DEFAULT IS RUN TESTS 0-10, 12, 13 & 15-21
001336	000000	OPNFLG: .WORD	0,0	:MODIFY TEST PARAMETER FLAGS
001342	000000	CLKSTA: .WORD	0	:CLOCK STATUS (0=NO CLOCK, +1=KW11-P, AND -1=KW11-L)
				:16 MILLISECONDS PER CLOCK TICK
				:16666 MICROSECONDS PER CLOCK TICK
001344	000020	TICKMS: .WORD	16.	
001346	040432	TICKUS: .WORD	16666.	
001350	000000	BYPASS: .WORD	0	:DRIVE UNDER TEST
001352	000000	CHKDRV: .WORD	0	:DRIVE MASK BIT
001354	000000	DRVMSK: .WORD	0	:STATUS/ERROR INDICATOR IS
001356	000000	SVSTAT: .WORD	0	:SAVED HERE ON AN ERROR
				:CYLINDER READ
001360	000000	CYL.RD: .WORD	0	:TRACK READ
001362	000000	TRK.RD: .WORD	0	:SECTOR READ
001364	000000	SEC.RD: .WORD	0	:CYLINDER DESIRED
001366	000000	CYL.DS: .WORD	0	:SECTOR DESIRED
001370	000000	SEC.DS: .WORD	0	:TRACK DESIRED
001372	000000	TRK.DS: .WORD	0	:CONTAINS THE LAST TRACK OF THE UNIT UNDER
001374	000000	LSTRK: .WORD	0	:TEST. RM02/3 = 4., RM05 = 18.
				:MINIMUM TIME
001376	000000	TIM.UP: .WORD	0	:NUMBER OF COUNTS BELOW MIN. LIMIT
001400	000000	.WORD	0	:MAXIMUM TIME
001402	000000	.WORD	0	:NUMBER OF COUNTS ABOVE MAX. LIMIT
001404	000000	.WORD	0	:TOTAL TIME OF ALL SEEKS
001406	000000	.WORD	0,0	:NUMBER OF SEEKS PERFORMED
001412	000000	.WORD	0	:MINIMUM TIME
001414	000000	TIM.DN: .WORD	0	:NUMBER OF COUNTS BELOW MIN. LIMIT
001416	000000	.WORD	0	:MAXIMUM TIME
001420	000000	.WORD	0	:NUMBER OF COUNTS ABOVE MAX. LIMIT
001422	000000	.WORD	0	:TOTAL TIME OF ALL SEEKS
001424	000000	.WORD	0,0	:NUMBER OF SEEKS PERFORMED
001430	000000	.WORD	0	:POINTS TO TABLE OF TIMES
001432	000000	TIM.PT: .WORD	0	:FATAL WRITE CHECK ERROR FLAG
001434	000000	WCEFLG: .WORD	0	:VARIABLE STALL
001436	000000	STALL0: .WORD	0	:SAVE DISK ADDRESS
001440	000000	SVADR: .WORD	0,0	:SEEK TIMER
001444	000000	SEKTMR: .WORD	0	:SEEK COUNTER
001446	000000	SEKCNT: .WORD	0	:TESTING RANGE FOR SERVO SETTLE DOWN TEST
001450	000000	DFLTA: .WORD	0	:WORD COUNT FOR A FULL TRACK IN 16 BIT MODE
001452	160000	TRCKWC: .WORD	-<256.*32.>	:10 MILLISECONDS STALL
001454	000012	STALL1: .WORD	10.	:10 MILLISECONDS STALL
001456	000012	STALL2: .WORD	10.	:5 SEC STALL
001460	011610	STALL3: .WORD	5000.	:MAX. INCREMENTING STALL ALLOWED IN TEST 4
001462	000031	MXSTAL: .WORD	25.	:NUMBER OF ERRORS ALLOWED IN TESTS 16 - 21
001464	020	ERR.CT: .BYTE	16.	:BEFORE GOING TO THE NEXT TEST
				:RESERVED
001465	000	.BYTE	0	:FLAG FOR DETECTING BAD SECTOR
001466	000000	BASFLG: .WORD	0	:THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
001470	000000	XXDP: .WORD	0	

001472

ERRCN: .BLKB 8.

;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
 ;'XXDP' DEVICE CODE FOR THE RM05/3/2.

;TOTAL ERROR COUNT FOR DRIVES 0-7.

001502 176700
 001504 000254 000240
 001510 000104 000106
 001514 172540
 001516 172542
 001520 172544
 001522 000100 000102
 001526 177546
 001530 177564
 001532 177566
 001534 177514
 001536 177516

;ADDRESSES AND VECTORS
 RH.ADR: .WORD 176700
 RHVEC: .WORD 254,240
 PKV: .WORD 104,106
 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

;RH/RM UNIBUS ADDRESS
 ;VECTOR ADDRESS AND PRIORITY
 ;KW11-P VECTOR ADDRESS
 ;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

001540 000001
 001542 000002
 001544 000004
 001546 000010
 001550 000020
 001552 000040
 001554 000100
 001556 000200
 001560 000400
 001562 001000
 001564 002000
 001566 004000
 001570 010000
 001572 020000
 001574 040000
 001576 100000
 001600 000001
 001602 000002
 001604 000004
 001606 000010
 001610 000020
 001612 000040
 001614 000100
 001616 000200

;BIT TABLE
 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

.SBTTL TIMING LIMITS

;ROTATIONAL TEST TABLES FOR RM05/3 DRIVES

;60HZ TABLE

001620 047754
 001622 000000
 001624 003142
 001626 003244

T7A: .WORD ROTATE
 .WORD 0
 .WORD 1634.
 .WORD 1700.

;LO LIMIT (16.67MS - 2%)
 ;HI LIMIT (16.67MS + 2%)

;50HZ TABLE

001630 047754
 001632 000000

T7B: .WORD ROTATE
 .WORD 0

001634	003131	.WORD	1625.	;LO LIMIT (16.67MS - 2.5%)
001636	003255	.WORD	1709.	;HI LIMIT (16.67MS + 2.5%)

;ROTATIONAL TEST TABLES FOR RM02 DRIVES

;60HZ TABLE				
001640	047754	T7A1: .WORD	ROTATE	
001642	000000	.WORD	0	
001644	004622	.WORD	2450.	;LO LIMIT (25.00MS - 2%)
001646	004766	.WORD	2550.	;HI LIMIT (25.00MS + 2%)

;50HZ TABLE				
001650	042754	T7B1: .WORD	ROTATE	
001652	000000	.WORD	0	
001654	004605	.WORD	2437.	;LO LIMIT (25.00MS - 2.5%)
001656	005003	.WORD	2563.	;HI LIMIT (25.00MS + 2.5%)

;SEEK TEST TABLES				
001660	050005	T10: .WORD	ONECYL	;FORWARD
001662	050152	.WORD	REV	;REVERSE
001664	000000	.WORD	0	;NO LO LIMIT
001666	001130	.WORD	600.	;HI LIMIT (6.0MS)

001670	050052	T11: .WORD	AVERAGE	;FORWARD
001672	050152	.WORD	REV	;REVERSE
001674	000000	.WORD	0	;NO LO LIMIT
001676	005670	.WORD	3000.	;HI LIMIT (30.0MS)

001700	050112	T12: .WORD	MXSEEK	;FORWARD
001702	050152	.WORD	REV	;REVERSE
001704	000000	.WORD	0	;NO LO LIMIT
001706	012574	.WORD	5500.	;HI LIMIT (55.0MS)

;SPECS. MESSAGE TABLES FOR ROTATIONAL AND TIMING TESTS

;ROTATIONAL MESSAGE AND LO/HI LIMITS FOR RM05/3 DRIVES

;60HZ TABLE				
001710	050350	SP7A: .WORD	MSG7XA	
001712	003142	.WORD	1634.	;LO LIMIT (16.67MS - 2%)
001714	003244	.WORD	1700.	;HI LIMIT (16.67MS + 2%)

;50HZ TABLE				
001716	050350	SP7B: .WORD	MSG7XA	
001720	003131	.WORD	1625.	;LO LIMIT (16.67MS - 2.5%)
001722	003255	.WORD	1709.	;HI LIMIT (16.67MS + 2.5%)

;ROTATIONAL MESSAGE AND LO/HI LIMITS FOR RM02 DRIVES

;60HZ TABLE				
001724	050425	SP7A1: .WORD	MSG7XB	
001726	004622	.WORD	2450.	;LO LIMIT (25.00MS - 2%)
001730	004766	.WORD	2550.	;HI LIMIT (25.00MS + 2%)

;50HZ TABLE				
001732	050425	SP7B1: .WORD	MSG7XB	
001734	004605	.WORD	2437.	;LO LIMIT (25.00MS - 2.5%)
001736	005003	.WORD	2563.	;HI LIMIT (25.00MS + 2.5%)

```

:TIMING TESTS MESSAGES AND LO/HI LIMITS
SP10: .WORD MSG10X
      .WORD 0 ;NO LO LIMIT
      .WORD 600. ;HI LIMIT (6.0MS)

SP11: .WORD MSG11X
      .WORD 0 ;NO LO LIMIT
      .WORD 3000. ;HI LIMIT (30.0MS)

SP12: .WORD MSG12X
      .WORD 0 ;NO LO LIMIT
      .WORD 5500. ;HI LIMIT (55.0MS)

:STATUS/ERROR MESSAGE POINTER TABLE
STATBL: .WORD MSGB14 ;OFFLINE OR UNSAFE DRIVE REQUESTED
        .WORD MSGB13 ;UNLOAD DRIVE REQUESTED
        .WORD MSGB12 ;PERSISTENT UNSAFE
        .WORD MSGB11 ;PARITY ERROR OCCURRED
        .WORD MSGB10 ;FATAL PARITY ERROR
        .WORD MSGB09 ;SOFTWARE TIMEOUT ON THIS DRIVE
        .WORD MSGB08 ;SOFTWARE TIMEOUT ON ANOTHER DRIVE
        .WORD MSGB06 ;ERROR OCCURRED DURING I/O OPERATION
        .WORD MSGB05 ;ERROR OCCURRED DURING NON-I/O OPERATION
        .WORD MSGB04 ;UNSAFE OCCURRED
        .WORD MSGB03 ;AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
        .WORD MSGB02 ;DRIVE HAS NOT RESPONDED TO PORT REQUEST
        .WORD MSGB01 ;DRIVE HAS BECOME NONEXISTENT

```

.SBTTL ERROR POINTER TABLE

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

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

002014

\$ERRTB:

;*ERROR 1

;* RH CONTROLLER INTERRUPT OCCURED (RMAS=0)
 ;* ERR PC RMAS
 ;* \$ERRPC \$REG3

11 002014 050660
 12 002016 052337
 13 002020 053714
 14 002022 054346

EM1
 DH1
 DT1
 DF1

;*ERROR 2

;* UNEXPECTED ATTENTION OCCURRED
 ;* ERR PC DRIVE RMAS RMD5 RMER1 RMMR2 RMER2
 ;* \$ERRPC \$REG1 \$REG3 RMERRS RMERRS+2 RMERRS+4 RMERRS+6

22 002024 050732
 23 002026 052354
 24 002030 053720
 25 002032 054352

EM2
 DH2
 DT2
 DF2

;*ERROR 3

;* MASSBUS PARITY ERROR (MCPE=1)
 ;* TEST ERR PC ADDRESS DATA
 ;* \$TMP0 \$ERRPC RD.ADR RD.WRD

33 002034 050770
 34 002036 052441
 35 002040 053736
 36 002042 054356

EM3
 DH3
 DT3
 DF3

;*ERROR 4

;* MASSBUS PARITY ERROR (PAR=1)
 ;* TEST ERR PC ADDRESS GDDATA BDDATA
 ;* \$TMP0 \$ERRPC WRT.ADR WRT.WD RD.WRD

44 002044 051025
 45 002046 052476

EM4
 DH4

```

46 002050 053746 DT4
47 002052 054362 DF4
48
49 :*ERROR 5
50
51 :* ADDRESS PLUG CHANGE BIT SET
52 :* ERR PC DRIVE RMAS RMD5 RMER1 RMMR2 RMER2
53 :* $ERRPC $REG1 $REG3 RMERRS RMERRS+2 RMERRS+4 RMERRS+6
54
55 002054 05106* EM5
56 002056 052354 DH2
57 002060 053720 DT2
58 002062 054352 DF2
59
60 :*ERROR 6 -- NOT USED
61
62 002064 000000 0
63 002066 000000 0
64 002070 000000 0
65 002072 000000 0
66
67 :*ERROR 7 -- NOT USED
68
69 002074 000000 0
70 002076 000000 0
71 002100 000000 0
72 002102 000000 0
73
74 :*ERROR 10
75
76 :* RH CONTROLLER FAILED TO RESPOND TO ADDRESSING
77 :* RMCS1 ERR PC
78 :* RH.ADR $ERRPC
79
80 002104 051115 EM10
81 002106 052545 DH10
82 002110 054000 DT10
83 002112 054366 DF10
84
85 :*ERROR 11
86
87 :* DRIVE SELECTED IS NOT ONLINE
88 :* DRIVE ERR PC
89 :* $REG2 $ERRPC
90
91 002114 051173 EM11
92 002116 052564 DH11
93 002120 054004 DT11
94 002122 054372 DF11
95
96 :*ERROR 12
97
98 :* IMPROPER HEADER DATA
99 :* TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR
100 :* $TMP0 $ERRPC $REG0 CHKDRV CYL.DS TRK.DS SEC.DS
101 :* GDCYL GDTRK GDSCTR BDCYL BDTRK BDSCTR
102 :* CYL.DS TRK.DS SEC.DS CYL.RD TRK.RD SEC.RD

```

103
 104
 105 002124 051230
 106 002126 052603
 107 002130 054010
 108 002132 054376
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119 002134 051255
 120 002136 052603
 121 002140 054042
 122 002142 054406
 123
 124
 125
 126
 127
 128 002144 000000
 129 002146 000000
 130 002150 054060
 131 002152 054416
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142 002154 051255
 143 002156 052603
 144 002160 054042
 145 002162 054406
 146
 147
 148
 149
 150
 151 002164 000000
 152 002166 000000
 153 002170 054060
 154 002172 054416
 155
 156
 157
 158
 159

```

;*  CYLNR, TRACK, AND SECTOR ARE DECIMAL
      EM12
      DH12
      DT12
      DF12

;*ERROR 13

;*  DATA COMPARE FAILURE
;*  TEST   ERR PC  TST PC  DRIVE  CYLNR  TRACK  SECTOR
;*  $TMP0  $ERRPC $REG0  CHKDRV  CYL.DS  TRK.DS  SEC.DS
;*        GDDAT  BDDAT  WRDCNT  GDADR  BDADR
;*        $GDDAT $BDDAT $REG4  $GDADR $BDADR
;*  CYLNR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
      EM13
      DH12
      DT13
      DF13

;*ERROR 14 -- FOLLOWS #13

;*        $GDDAT $BDDAT $REG4  $GDADR $BDADR
      0
      0
      DT13A
      DF14

;*ERROR 15

;*  DATA COMPARE FAILURE
;*  TEST   ERR PC  TST PC  DRIVE  CYLNR  TRACK  SECTOR
;*  $TMP0  $ERRPC $REG0  CHKDRV  CYL.DS  TRK.DS  SEC.DS
;*        GDDAT  BDDAT  WRDCNT  GDADR  BDADR
;*        $GDDAT $BDDAT $REG4  $GDADR $BDADR
;*  CYLNR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
      EM13
      DH12
      DT13
      DF13

;*ERROR 16 -- FOLLOWS #15

;*  $GDDAT $BDDAT $REG4  $GDADR $BDADR
      0
      0
      DT13A
      DF14

;*ERROR 17

;*  DISK ERROR IN TIMING TEST
;*  TEST   ERR PC  DRIVE  RMCS1  RMD5  RMER1  RMMR2  RMER2

```

```

160      ;* $TMP0 $ERRPC CHKDRV RM.REG RM.REG+12 RM.REG+14 RM.REG+40 RM.REG+42
161
162      EM17
163      DH17
164      DT17
165      DF17
166
167      ;*ERROR 20
168
169      ;* CLOCK (KW11-?) OVERFLOW IN TIMING TEST
170      ;* TEST ERR PC DRIVE RMCS1 RMD5 RMER1 RMMR2 RMER2
171      ;* $TMP0 $ERRPC CHKDRV RM.REG RM.REG+12 RM.REG+14 RM.REG+40 RM.REG+42
172
173      EM20
174      DH17
175      DT17
176      DF17
177
178      ;*ERROR 21
179
180      ;* DATA COMPARE FAILURE
181      ;* TEST ERR PC TST PC DRIVE CYLNDR TRACK
182      ;* $TMP0 $ERRPC $REG0 CHKDRV CYL.DS TRK.DS
183      ;* GDDAT BDDAT WRDCNT SECTOR
184      ;* $REG1 $BDDAT $REG4 $REG1
185      ;* CYLINDR, TRACK, WRDCNT, AND SECTOR ARE DECIMAL
186
187      EM13
188      DH21
189      DT21
190      DF21
191
192      ;*ERROR 22--FOLLOWS #21
193
194      ;* $REG1 $BDDAT $REG4 $REG1
195
196      0
197      0
198      DT21A
199      DF22
200
201      ;*ERROR 23
202
203      ;* DISK ERROR DURING SEEK
204      ;* TEST ERR PC DRIVE CYLNDR RMCS1 RMCS2 RMD5
205      ;* $TMP0 $ERRPC CHKDRV CYL.DS RM.REG RM.REG+10 RM.REG+12
206      ;* RMER1 RMMR2 RMER2 RMD5 RMHR
207      ;* RM.REG+14 RM.REG+40 RM.REG+42 RM.REG+34 RM.REG+36
208
209      EM23
210      DH23
211      DT23
212      DF23
213
214      ;*ERROR 24
215
216      ;* SEEK NOT COMPLETE WITHIN 120 MS
  
```

```

217      ;* TEST      ERR PC  DRIVE  CYLNDR  RMCS1  RMCS2  RMD5
218      ;* $TMP0      $ERRPC  CHKDRV  CYL.DS  RM.REG  RM.REG+10 RM.REG+12
219      ;*          RMER1      RMMR2      RMER2      RMDC      RMHR
220      ;*          RM.REG+14 RM.REG+40 RM.REG+42 RM.REG+34 RM.REG+36
221
222      002244  051432      EM24
223      002246  053231      DH23
224      002250  054136      DT23
225      002252  054442      DF23
226
227      ;*****
228      ;* ERRORS 23-40 NOT USED
229      ;* ERRORS 41-46 WILL HAVE AN EM THAT
230
231      ;* VARIES DEPENDING ON THE ERROR, IT WILL BE IN THE FORM:
232      ;* RH/RM ERROR (MESSAGE)
233      ;* WHERE MESSAGE WILL BE ONE OR MORE OF THE FOLLOWING:
234      ;* 1) OFFLINE OR UNSAFE DRIVE REQUESTED
235      ;* 2) UNLOADED DRIVE REQUESTED
236      ;* 3) PERSISTENT UNSAFE
237      ;* 4) PARITY ERROR OCCURRED
238      ;* 5) FATAL PARITY ERROR
239      ;* 6) SOFTWARE TIMEOUT ON THIS DRIVE
240      ;* 7) SOFTWARE TIMEOUT ON ANOTHER DRIVE
241      ;* 8) ERROR OCCURRED DURING I/O OPERATION
242      ;* 9) ERROR OCCURRED DURING NON-I/O OPERATION
243      ;* 10) UNSAFE OCCURRED
244      ;* 11) AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
245      ;*****
246
247      002254      ITEM41:
248
249      ;*ERROR 41
250
251      ;* RH/RM ERROR (MESSAGE)
252      ;* TEST      ERR PC  TST PC  DRIVE
253      ;* $TMP0      $ERRPC  $REGO  CHKDRV
254
255      002254  051472      EM41
256      002256  053363      DH41
257      002260  054166      DT41
258      002262  054452      DF41
259
260      ;*ERROR 42
261
262      ;* RH/RM ERROR (MESSAGE)
263      ;* TEST      ERR PC  TST PC  DRIVE  RMCS1  RMCS2  RMD5
264      ;* $TMP0      $ERRPC  $REGO  CHKDRV  RM.REG  RM.REG+10 RM.REG+12
265
266      002264  051472      EM41
267      002266  053421      DH42
268      002270  054176      DT42
269      002272  054456      DF42
270
271      ;*ERROR 43
272
273      ;* RH/RM ERROR (MESSAGE)

```

```

274      : * TEST      ERR PC  TST PC  DRIVE  RMCS1  RMCS2  RMDS
275      : * $TMP0    $ERRPC  $REG0  CHKDRV  RM.REG  RM.REG+10 RM.REG+12
276      : *          RMER1    RMMR2    RMER2
277      : *          RM.REG+14 RM.REG+40 RM.REG+42
278
279 002274 051472      EM41
280 002276 053421      DH42
281 002300 054214      DT43
282 002302 054462      DF43
283
284      : *ERROR 44
285
286      : * RH/RM ERROR (MESSAGE)
287      : * TEST      ERR PC  TST PC  DRIVE  CYLNDR  TRACK  SECTOR
288      : * $TMP0    $ERRPC  $REG0  CHKDRV  CYL.DS  TRK.DS  SEC.DS
289      : *          RMCS1    RMCS2    RMDS    RMHR    RMDC    RMDA
290      : *          RM.REG    RM.REG+10 RM.REG+12 RM.REG+36 RM.REG+34 RM.REG+06
291      : *          RMER1    RMMR2    RMER2
292      : *          RM.REG+14 RM.REG+40 RM.REG+42
293      : *          CYLNDR, TRACK, AND SECTOR ARE DECIMAL
294
295 002304 051472      EM41
296 002306 052603      DH12
297 002310 054240      DT44
298 002312 054472      DF44
299
300      : *ERROR 45
301
302      : * RH/RM ERROR (MESSAGE)
303      : * TEST      ERR PC  TST PC  DRIVE  CYLNDR  TRACK  SECTOR
304      : * $TMP0    $ERRPC  $REG0  CHKDRV  CYL.DS  TRK.DS  SEC.DS
305      : *          RMCS1    RMCS2    RMDS    RMHR    RMDC    RMDA
306      : *          RM.REG    RM.REG+10 RM.REG+12 RM.REG+36 RM.REG+34 RM.REG+06
307      : *          RMER1    RMMR2    RMER2    RMWC    RMBA    RMDB
308      : *          RM.REG+14 RM.REG+40 RM.REG+42 RM.REG+2  RM.REG+4 RM.REG+22
309      : *          CYLNDR, TRACK, AND SECTOR ARE DECIMAL
310
311 002314 051472      EM41
312 002316 052603      DH12
313 002320 054300      DT45
314 002322 054506      DF45
315
316      : *ERROR 46
317
318      : * FATAL WRITE CHECK ERROR (MESSAGE)
319      : * TEST      ERR PC  TST PC  DRIVE  CYLNDR  TRACK  SECTOR
320      : * $TMP0    $ERRPC  $REG0  CHKDRV  CYL.DS  TRK.DS  SEC.DS
321      : *          RMCS1    RMCS2    RMDS    RMHR    RMDC    RMDA
322      : *          RM.REG    RM.REG+10 RM.REG+12 RM.REG+36 RM.REG+34 RM.REG+06
323      : *          RMER1    RMMR2    RMER2    RMWC    RMBA    RMDB
324      : *          RM.REG+14 RM.REG+40 RM.REG+42 RM.REG+2  RM.REG+4 RM.REG+22
325      : *          CYLNDR, TRACK, AND SECTOR ARE DECIMAL
326
327 002324 051506      EM46
328 002326 052603      DH12
329 002330 054300      DT45
330 002332 054506      DF45

```

```

1
2
3
4 002334 000000
5
6 002336 000000
7 002340 000000
8 002342 000000
9 002344 000000
10 002346 000000
11 002350 000000
12 002352 000000
13 002354 000000
14 002356 000000
15 002360 000000
16 002362 000000 000000 000000
17
18
19 002370 000000
20 002372 000000
21
22
23 002374 003132
26 002376 003150
    002400 003176
    002402 003216
    002404 003236
    002406 003256
    002410 003276
    002412 003316
    002414 003336
    002416 003354
    002420 003374
    002422 003410
    002424 003420
    002426 003430
    002430 003440
    002432 003452
    002434 003462
    002436 003516
    002440 003526
27 002442 000000
28
29
30 002444 032767
31 002446 001466
32 002450 001466
33 002452 001466
34 002454 000004
35 002456 000004
36 002460 000004
37 002462 000022
38 002464 000022
39 002466 000022
40 002470 000037
41 002472 000037
42 002474 177777

.SBTTL TEST PARAMETER POINTERS AND TABLES

;COMMON STORAGE FOR TEST PARAMETERS
PRM: .WORD 0 ;THIS WORD TELLS WHICH OF THE
;FOLLOWING PARAMETERS ARE TO BE USED
RPT: .WORD 0 ;REPEAT COUNTS FOR ALL TESTS
FC: .WORD 0 ;FIRST CYLINDER
LC: .WORD 0 ;LAST CYLINDER
IC: .WORD 0 ;INCREMENT CYLINDER
FT: .WORD 0 ;FIRST TRACK
LT: .WORD 0 ;LAST TRACK
IT: .WORD 0 ;INCREMENT TRACK
FS: .WORD 0 ;FIRST SECTOR
LS: .WORD 0 ;LAST SECTOR
PAT: .WORD 0 ;PATTERN CODE
      .WORD 0,0,0 ;FILLER WORDS FOR COMMON TABLE USED BY THE
                  ;'OPNTST' ROUTINE.

NC1: .WORD 0 ;NEW CYLINDER ADDRESS
NC2: .WORD 0 ;NEW CYLINDER ADDRESS

;TABLE OF PARAMETER POINTERS
PRMPT: .WORD PRM0
      .WORD PRM1
      .WORD PRM2
      .WORD PRM3
      .WORD PRM4
      .WORD PRM5
      .WORD PRM6
      .WORD PRM7
      .WORD PRM10
      .WORD PRM11
      .WORD PRM12
      .WORD PRM13
      .WORD PRM14
      .WORD PRM15
      .WORD PRM16
      .WORD PRM17
      .WORD PRM20
      .WORD PRM21
      .WORD PRM22
      .WORD 0 ;TERMINATOR

;TABLE OF PARAMETER UPPER LIMITS
PRMLMT: .WORD 32767 ;'R'
      .WORD 822. ;'FC'
      .WORD 822. ;'LC'
      .WORD 822. ;'IC'
      .WORD 4 ;'FT'
      .WORD 4 ;'LT'
      .WORD 4 ;'IT'
      .WORD 18. ;'FT'
      .WORD 18. ;'LT'
      .WORD 18. ;'IT'
      .WORD 31. ;'FS'
      .WORD 31. ;'LS'
      .WORD 177777 ;'PAT'

```

43

44

45 002476 047210

46 002500 047212

47 002502 047215

48 002504 047220

49 002506 047223

50 002510 047226

51 002512 047231

52 002514 047234

53 002516 047240

54 002520 047244

55 002522 047250

56 002524 047253

57

58

59

60

61 002526 002227 000310 000000

62 002544 006677 000144 000000

63 002572 002237 000001 000000

64 002612 002237 000010 000000

65 002632 002237 000001 000000

66 002652 002237 000001 000000

67 002672 002237 000001 000000

68 002712 000667 011610 000000

69 002732 000237 000001 000000

70 002750 002237 000001 000000

71 002770 002223 000001 000000

72 003004 000007 000001 000000

73 003014 000007 000001 000000

74 003024 000007 000001 000000

75 003034 000223 000001 000000

76 003046 002003 000001 000000

77 003056 017777 000001 000000

78 003112 000007 001750 000000

79 003122 000007 011610 000000

80

81

82

83

84 003132 002227

85 003134 000310

86 003136 000000

87 003140 001466

88 003142 000000

89 003144 000000

90 003146 000000

91

92

93 003150 006677

94 003152 000144

95 003154 000000

96 003156 000400

97 003160 000000

98 003162 000000

99 003164 000000

TABLE OF PARAMETER MESSAGE POINTERS

PRMSG: .WORD MSG.R
 .WORD MSG.FC
 .WORD MSG.LC
 .WORD MSG.IC
 .WORD MSG.FT
 .WORD MSG.LT
 .WORD MSG.IT
 .WORD MES.FT
 .WORD MES.LT
 .WORD MES.IT
 .WORD MSG.FS
 .WORD MSG.LS

;STATUS/ERROR INDICATOR MESSAGES POINTER TABLE

;DEFAULT VALUES OF TEST PARAMETERS

DFLT: .WORD 2227,200.,0,822.,0,0,0 ;RECAL/RANDOM SEEK (T0)
 .WORD 6677,100.,0,256.,0,0,0,0,0,0 ;SEEK/SEEK (T1)
 .WORD 2237,1,0,822.,1,0,0,0 ;INCREMENT SEEK (T2)
 .WORD 2237,10,0,512.,1,0,0,0 ;STEPPING SEEK (T3)
 .WORD 2237,1,0,822.,1,0,0,0 ;OSCILLATING SEEK (T4)
 .WORD 2237,1,0,822.,1,0,0,0 ;CONVERGING/DIVERGING SEEK (T5)
 .WORD 2237,1,0,822.,1,0,0,0 ;SERVO ADDRESSING LOGIC NOISE (T6)
 .WORD 667,5000.,0,822.,0,4,0,18. ;RANDOM SEEK TEST (T7)
 .WORD 237,1,0,822.,100.,0,0 ;SERVO SETTLE DOWN TEST (T10)
 .WORD 2237,1,0,822.,1,0,0,0 ;ALL SEEKS TEST (T11)
 .WORD 2223,1,0,0,0,0 ;ROTATIONAL SPEED TIMING TEST (T12)
 .WORD 7,1,0,822. ;ONE CYLINDER SEEK TIMING TEST (T13)
 .WORD 7,1,0,220. ;AVERAGE SEEK TIMING TEST (T14)
 .WORD 7,1,0,822. ;MAXIMUM SEEK TIMING TEST (T15)
 .WORD 223,1,0,0,0 ;SECTOR ADDRESSING TEST (T16)
 .WORD 2003,1,0,0 ;TRACK ADDRESSING TEST (T17)
 .WORD 17777,1,0,821.,64.,0,4,1,0,18.,1,1,0,17777 ;DATA TEST (T20)
 .WORD 7,1000.,0,821. ;EXERCISER (T21)
 .WORD 7,5000.,0,255. ;SEEK TIME ADJUSTMENT TEST (T22)

;PARAMETER TABLES

;RECAL/RANDOM SEEK (T0)

PRMO: .WORD 2227
 .WORD 200.
 .WORD 0
 .WORD 822.
 .WORD 0
 .WORD 0
 .WORD 0

;SEEK/SEEK (T1)

PRM1: .WORD 6677
 .WORD 100.
 .WORD 0
 .WORD 256.
 .WORD 0
 .WORD 0
 .WORD 0

100	003166	000000	.WORD	0
101	003170	000000	.WORD	0
102	003172	000000	.WORD	0
103	003174	000000	.WORD	0
104				
105			: INCREMENT SEEK (T2)	
106	003176	002237	PRM2: .WORD	2237
107	003200	000001	.WORD	1
108	003202	000000	.WORD	0
109	003204	001466	.WORD	822.
110	003206	000001	.WORD	1
111	003210	000000	.WORD	0
112	003212	000000	.WORD	0
113	003214	000000	.WORD	0
114				
115			: STEPPING SEEK (T3)	
116	003216	002237	PRM3: .WORD	2237
117	003220	000001	.WORD	1
118	003222	000000	.WORD	0
119	003224	001000	.WORD	512.
120	003226	000001	.WORD	1
121	003230	000000	.WORD	0
122	003232	000000	.WORD	0
123	003234	000000	.WORD	0
124				
125			: OSCILLATING SEEK (T4)	
126	003236	002237	PRM4: .WORD	2237
127	003240	000001	.WORD	1
128	003242	000000	.WORD	0
129	003244	001466	.WORD	822.
130	003246	000001	.WORD	1
131	003250	000000	.WORD	0
132	003252	000000	.WORD	0
133	003254	000000	.WORD	0
134				
135			: CONVERGING/DIVERGING SEEK (T5)	
136	003256	002237	PRM5: .WORD	2237
137	003260	000001	.WORD	1
138	003262	000000	.WORD	0
139	003264	001466	.WORD	822.
140	003266	000001	.WORD	1
141	003270	000000	.WORD	0
142	003272	000000	.WORD	0
143	003274	000000	.WORD	0
144				
145			: SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)	
146	003276	002237	PRM6: .WORD	2237
147	003300	000001	.WORD	1
148	003302	000000	.WORD	0
149	003304	001466	.WORD	822.
150	003306	000001	.WORD	1
151	003310	000000	.WORD	0
152	003312	000000	.WORD	0
153	003314	000000	.WORD	0
154				
155			: RANDOM SEEK TEST (T7)	
156	003316	000667	PRM7: .WORD	667

157	003320	011610	.WORD	5000.
158	003322	000000	.WORD	0
159	003324	001466	.WORD	822.
160	003326	000000	.WORD	0
161	003330	000004	.WORD	4
162	003332	000000	.WORD	0
163	003334	000022	.WORD	18.
164				
165			;SERVO SETTLE DOWN TEST (T10)	
166	003336	000237	PRM10: .WORD	237
167	003340	000001	.WORD	1
168	003342	000000	.WORD	0
169	003344	001466	.WORD	822.
170	003346	000144	.WORD	100.
171	003350	000000	.WORD	0
172	003352	000000	.WORD	0
173				
174			;ALL SEEKS TEST (T11)	
175	003354	002237	PRM11: .WORD	2237
176	003356	000001	.WORD	1
177	003360	000000	.WORD	0
178	003362	001466	.WORD	822.
179	003364	000001	.WORD	1
180	003366	000000	.WORD	0
181	003370	000000	.WORD	0
182	003372	000000	.WORD	0
183				
184			;ROTATIONAL SPEED TIMING TEST (T12)	
185	003374	002223	PRM12: .WORD	2223
186	003376	000001	.WORD	1
187	003400	000000	.WORD	0
188	003402	000000	.WORD	0
189	003404	000000	.WORD	0
190	003406	000000	.WORD	0
191				
192			;ONE CYLINDER SEEK TIMING TEST (T13)	
193	003410	000007	PRM13: .WORD	7
194	003412	000001	.WORD	1
195	003414	000000	.WORD	0
196	003416	001466	.WORD	822.
197				
198			;AVERAGE SEEK TIMING TEST (T14)	
199	003420	000007	PRM14: .WORD	7
200	003422	000001	.WORD	1
201	003424	000000	.WORD	0
202	003426	000334	.WORD	220.
203				
204			;MAXIMUM SEEK TIMING TEST (T15)	
205	003430	000007	PRM15: .WORD	7
206	003432	000001	.WORD	1
207	003434	000000	.WORD	0
208	003436	001466	.WORD	822.
209				
210			;SECTOR ADDRESSING TEST (T16)	
211	003440	000223	PRM16: .WORD	223
212	003442	000001	.WORD	1
213	003444	000000	.WORD	0

```
214 003446 000000          .WORD 0
215 003450 000000          .WORD 0
216
217          ;TRACK ADDRESSING TEST (T17)
218 003452 002003          PRM17: .WORD 2003
219 003454 000001          .WORD 1
220 003456 000000          .WORD 0
221 003460 000000          .WORD 0
222
223          ;DATA TEST (T20)
224 003462 017777          PRM20: .WORD 17777
225 003464 000001          .WORD 1
226 003466 000000          .WORD 0
227 003470 001465          .WORD 821.
228 003472 000100          .WORD 64.
229 003474 000000          .WORD 0
230 003476 000004          .WORD 4
231 003500 000001          .WORD 1
232 003502 000000          .WORD 0
233 003504 000022          .WORD 18.
234 003506 000001          .WORD 1
235 003510 000001          .WORD 1
236 003512 000000          .WORD 0
237 003514 177777          PTRN15: .WORD 177777
238
239          ;EXERCISER (T21)
240 003516 000007          PRM21: .WORD 7
241 003520 001750          .WORD 1000.
242 003522 000000          .WORD 0
243 003524 001465          .WORD 821.
244
245          ;SEEK TIME ADJUSTMENT TEST (T22)
246 003526 000007          PRM22: .WORD 7
247 003530 011610          .WORD 5000.
248 003532 000000          .WORD 0
249 003534 000377          .WORD 255.
```

```

1      .SBTTL DATA PATTERN POINTERS AND TABLES
2
3 003536 003576      PAT.PT: .WORD PAT0      ;DATA PATTERN 0
6 003540 003636      .WORD PAT1      ;DATA PATTERN 1
  003542 003676      .WORD PAT2      ;DATA PATTERN 2
  003544 003736      .WORD PAT3      ;DATA PATTERN 3
  003546 003776      .WORD PAT4      ;DATA PATTERN 4
  003550 004036      .WORD PAT5      ;DATA PATTERN 5
  003552 004076      .WORD PAT6      ;DATA PATTERN 6
  003554 004136      .WORD PAT7      ;DATA PATTERN 7
  003556 004176      .WORD PAT8      ;DATA PATTERN 8
  003560 004236      .WORD PAT9      ;DATA PATTERN 9
  003562 004276      .WORD PAT10     ;DATA PATTERN 10
  003564 004336      .WORD PAT11     ;DATA PATTERN 11
  003566 004376      .WORD PAT12     ;DATA PATTERN 12
  003570 004436      .WORD PAT13     ;DATA PATTERN 13
  003572 004476      .WORD PAT14     ;DATA PATTERN 14
  003574 004536      .WORD PAT15     ;DATA PATTERN 15
7
8 003576 155555      PAT0: .WORD 155555      ;PATTERN 0 (WORST CASE)
9 003600 133333      .WORD 133333
10 003602 155555      .WORD 155555
11 003604 133333      .WORD 133333
12 003606 155555      .WORD 155555
13 003610 133333      .WORD 133333
14 003612 155555      .WORD 155555
15 003614 133333      .WORD 133333
16 003616 155555      .WORD 155555
17 003620 133333      .WORD 133333
18 003622 155555      .WORD 155555
19 003624 133333      .WORD 133333
20 003626 155555      .WORD 155555
21 003630 133333      .WORD 133333
22 003632 155555      .WORD 155555
23 003634 133333      .WORD 133333
24
25 003636 000001      PAT1: .WORD 000001      ;PATTERN 1
26 003640 000003      .WORD 000003
27 003642 000007      .WORD 000007
28 003644 000017      .WORD 000017
29 003646 000037      .WORD 000037
30 003650 000077      .WORD 000077
31 003652 000177      .WORD 000177
32 003654 000377      .WORD 000377
33 003656 000777      .WORD 000777
34 003660 001777      .WORD 001777
35 003662 003777      .WORD 003777
36 003664 007777      .WORD 007777
37 003666 017777      .WORD 017777
38 003670 037777      .WORD 037777
39 003672 077777      .WORD 077777
40 003674 177777      .WORD 177777
41
42 003676 177776      PAT2: .WORD 177776      ;PATTERN 2
43 003700 177774      .WORD 177774
44 003702 177770      .WORD 177770
45 003704 177760      .WORD 177760

```

46	003706	177740	.WORD	177740	
47	003710	177700	.WORD	177700	
48	003712	177600	.WORD	177600	
49	003714	177400	.WORD	177400	
50	003716	177000	.WORD	177000	
51	003720	176000	.WORD	176000	
52	003722	174000	.WORD	174000	
53	003724	170000	.WORD	170000	
54	003726	160000	.WORD	160000	
55	003730	140000	.WORD	140000	
56	003732	100000	.WORD	100000	
57	003734	000000	.WORD	000000	
58					
59	003736	000000	PAT3: .WORD	000000	:PATTERN 3
60	003740	000000	.WORD	000000	
61	003742	000000	.WORD	000000	
62	003744	177777	.WORD	177777	
63	003746	177777	.WORD	177777	
64	003750	177777	.WORD	177777	
65	003752	000000	.WORD	000000	
66	003754	000000	.WORD	000000	
67	003756	177777	.WORD	177777	
68	003760	177777	.WORD	177777	
69	003762	000000	.WORD	000000	
70	003764	177777	.WORD	177777	
71	003766	000000	.WORD	000000	
72	003770	177777	.WORD	177777	
73	003772	000000	.WORD	000000	
74	003774	177777	.WORD	177777	
75					
76	003776	133331	PAT4: .WORD	133331	:PATTERN 4
77	004000	133331	.WORD	133331	
78	004002	133331	.WORD	133331	
79	004004	133331	.WORD	133331	
80	004006	133331	.WORD	133331	
81	004010	133331	.WORD	133331	
82	004012	133331	.WORD	133331	
83	004014	133331	.WORD	133331	
84	004016	133331	.WORD	133331	
85	004020	133331	.WORD	133331	
86	004022	133331	.WORD	133331	
87	004024	133331	.WORD	133331	
88	004026	133331	.WORD	133331	
89	004030	133331	.WORD	133331	
90	004032	133331	.WORD	133331	
91	004034	133331	.WORD	133331	
92					
93	004036	052525	PAT5: .WORD	052525	:PATTERN 5
94	004040	052525	.WORD	052525	
95	004042	052525	.WORD	052525	
96	004044	125252	.WORD	125252	
97	004046	125252	.WORD	125252	
98	004050	125252	.WORD	125252	
99	004052	052525	.WORD	052525	
100	004054	052525	.WORD	052525	
101	004056	125252	.WORD	125252	
102	004060	125252	.WORD	125252	

103	004062	052525	.WORD	052525	
104	004064	125252	.WORD	125252	
105	004066	052525	.WORD	052525	
106	004070	125252	.WORD	125252	
107	004072	052525	.WORD	052525	
108	004074	125252	.WORD	125252	
109					
110	004076	155555	PAT6: .WORD	155555	;PATTERN 6
111	004100	155555	.WORD	155555	
112	004102	155555	.WORD	155555	
113	004104	155555	.WORD	155555	
114	004106	155555	.WORD	155555	
115	004110	155555	.WORD	155555	
116	004112	155555	.WORD	155555	
117	004114	155555	.WORD	155555	
118	004116	155555	.WORD	155555	
119	004120	155555	.WORD	155555	
120	004122	155555	.WORD	155555	
121	004124	155555	.WORD	155555	
122	004126	155555	.WORD	155555	
123	004130	155555	.WORD	155555	
124	004132	155555	.WORD	155555	
125	004134	155555	.WORD	155555	
126					
127	004136	026455	PAT7: .WORD	026455	;PATTERN 7
128	004140	026455	.WORD	026455	
129	004142	026455	.WORD	026455	
130	004144	151322	.WORD	151322	
131	004146	151322	.WORD	151322	
132	004150	151322	.WORD	151322	
133	004152	026455	.WORD	026455	
134	004154	026455	.WORD	026455	
135	004156	151322	.WORD	151322	
136	004160	151322	.WORD	151322	
137	004162	026455	.WORD	026455	
138	004164	151322	.WORD	151322	
139	004166	026455	.WORD	026455	
140	004170	151322	.WORD	151322	
141	004172	026455	.WORD	026455	
142	004174	151322	.WORD	151322	
143					
144	004176	155555	PAT8: .WORD	155555	;PATTERN 8 (WORST CASE)
145	004200	133333	.WORD	133333	
146	004202	155555	.WORD	155555	
147	004204	133333	.WORD	133333	
148	004206	155555	.WORD	155555	
149	004210	133333	.WORD	133333	
150	004212	155555	.WORD	155555	
151	004214	133333	.WORD	133333	
152	004216	155555	.WORD	155555	
153	004220	133333	.WORD	133333	
154	004222	155555	.WORD	155555	
155	004224	133333	.WORD	133333	
156	004226	155555	.WORD	155555	
157	004230	133333	.WORD	133333	
158	004232	155555	.WORD	155555	
159	004234	133333	.WORD	133333	

160				
161	004236	000001	PAT9: .WORD	000001 ;PATTERN 9
162	004240	000002	.WORD	000002
163	004242	000004	.WORD	000004
164	004244	000010	.WORD	000010
165	004246	000020	.WORD	000020
166	004250	000040	.WORD	000040
167	004252	000100	.WORD	000100
168	004254	000200	.WORD	000200
169	004256	000400	.WORD	000400
170	004260	001000	.WORD	001000
171	004262	002000	.WORD	002000
172	004264	004000	.WORD	004000
173	004266	010000	.WORD	010000
174	004270	020000	.WORD	020000
175	004272	040000	.WORD	040000
176	004274	100000	.WORD	100000
177				
178	004276	177776	PAT10: .WORD	177776 ;PATTERN 10
179	004300	177775	.WORD	177775
180	004302	177773	.WORD	177773
181	004304	177767	.WORD	177767
182	004306	177757	.WORD	177757
183	004310	177737	.WORD	177737
184	004312	177677	.WORD	177677
185	004314	177577	.WORD	177577
186	004316	177377	.WORD	177377
187	004320	176777	.WORD	176777
188	004322	175777	.WORD	175777
189	004324	173777	.WORD	173777
190	004326	167777	.WORD	167777
191	004330	157777	.WORD	157777
192	004332	137777	.WORD	137777
193	004334	077777	.WORD	077777
194				
195	004336	172666	PAT11: .WORD	172666 ;PATTERN 11
196	004340	155555	.WORD	155555
197	004342	172666	.WORD	172666
198	004344	155555	.WORD	155555
199	004346	172666	.WORD	172666
200	004350	155555	.WORD	155555
201	004352	172666	.WORD	172666
202	004354	155555	.WORD	155555
203	004356	172666	.WORD	172666
204	004360	155555	.WORD	155555
205	004362	172666	.WORD	172666
206	004364	155555	.WORD	155555
207	004366	172666	.WORD	172666
208	004370	155555	.WORD	155555
209	004372	172666	.WORD	172666
210	004374	155555	.WORD	155555
211				
212	004376	077777	PAT12: .WORD	077777 ;PATTERN 12
213	004400	137777	.WORD	137777
214	004402	157777	.WORD	157777
215	004404	167777	.WORD	167777
216	004406	173777	.WORD	173777

217	004410	175777	.WORD	175777	
218	004412	176777	.WORD	176777	
219	004414	177377	.WORD	177377	
220	004416	177577	.WORD	177577	
221	004420	177677	.WORD	177677	
222	004422	177737	.WORD	177737	
223	004424	177757	.WORD	177757	
224	004426	177767	.WORD	177767	
225	004430	177773	.WORD	177773	
226	004432	177775	.WORD	177775	
227	004434	177776	.WORD	177776	
228					
229	004436	153333	PAT13: .WORD	153333	;PATTERN 13
230	004440	066667	.WORD	066667	
231	004442	153333	.WORD	153333	
232	004444	066667	.WORD	066667	
233	004446	153333	.WORD	153333	
234	004450	066667	.WORD	066667	
235	004452	153333	.WORD	153333	
236	004454	066667	.WORD	066667	
237	004456	153333	.WORD	153333	
238	004460	066667	.WORD	066667	
239	004462	153333	.WORD	153333	
240	004464	066667	.WORD	066667	
241	004466	153333	.WORD	153333	
242	004470	066667	.WORD	066667	
243	004472	153333	.WORD	153333	
244	004474	066667	.WORD	066667	
245					
246	004476	000000	PAT14: .WORD	000000	;PATTERN 14
247	004500	177777	.WORD	177777	
248	004502	177777	.WORD	177777	
249	004504	177777	.WORD	177777	
250	004506	177777	.WORD	177777	
251	004510	177777	.WORD	177777	
252	004512	177777	.WORD	177777	
253	004514	177777	.WORD	177777	
254	004516	177777	.WORD	177777	
255	004520	177777	.WORD	177777	
256	004522	177777	.WORD	177777	
257	004524	177777	.WORD	177777	
258	004526	177777	.WORD	177777	
259	004530	177777	.WORD	177777	
260	004532	177777	.WORD	177777	
261	004534	177777	.WORD	177777	
262					
263	004536	177777	PAT15: .WORD	177777	;PATTERN 15
264	004540	000000	.WORD	000000	
265	004542	000000	.WORD	000000	
266	004544	000000	.WORD	000000	
267	004546	000000	.WORD	000000	
268	004550	000000	.WORD	000000	
269	004552	000000	.WORD	000000	
270	004554	000000	.WORD	000000	
271	004556	000000	.WORD	000000	
272	004560	000000	.WORD	000000	
273	004562	000000	.WORD	000000	

274	004564	000000	.WORD	000000
275	004566	000000	.WORD	000000
276	004570	000000	.WORD	000000
277	004572	000000	.WORD	000000
278	004574	000000	.WORD	000000

```

1      ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS
2
3      004576 011600      BADTMO: MOV      (SP),R0      ;SAVE PC WHERE THE TIME OUT OCCURED
4      004600 005740      TST      -(R0)      ;ADJUST PC -2
5      004602 022626      CMP      (SP)+,(SP)+    ;RESTORE STACK POINTER
6      004604 104401 004612  TYPE      65$      ;TYPE ASCII STRING
          004610 000417      BR       64$      ;GET OVER THE ASCII
          ;:65$: .ASCIIZ <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
          64$:
          MOV      R0,-(SP)      ;SETUP FOR TYPING OUT PC
          TYP0C
          NOP
          ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
          ;TO STOP ON UNEXPECTED TIMEOUT.
          BR      START1      ;BRANCH TO START1
12
13      .SBTTL START OF PROGRAM
14
15      004660 012737 177777 001324  START3: MOV      #-1,BUSADR      ;GET BUSADR FLAG
16      004666 000402      BR      STRT1A
17
18      004670 005037 001324      START1: CLR      BUSADR      ;CLR BUSADR FLAG
19      004674 005037 001322  STRT1A: CLR      CNTRLC      ;NO CONTROL 'C'
20      004700 000411      BR      START
21
22      004702 012737 177777 001324  START4: MOV      #-1,BUSADR      ;SET BUSADR FLAG
23      004710 000402      BR      STRT2A
24
25      004712 005037 001324      START2: CLR      BUSADR      ;CLR BUSADR FLAG
26      004716 012737 177777 001322  STRT2A: MOV      #-1,CNTRLC      ;SET CONTROL 'C' FLAG
27
28      004724 000240      START:  NOP
29      004726 005227 000000      INC      #0      ;TTY LOOP, WAIT FOR INCREMENT
30      004732 001375      BNE      -4      ;OF WORD
31      004734 000005      RESET      ;CLEAR THE WORLD
32
33      .SBTTL INITIALIZE THE COMMON TAGS
          ;:CLEAR THE COMMON TAGS ($CMTAG) AREA
          MOV      #SCMTAG,R6      ;:FIRST LOCATION TO BE CLEARED
          CLR      (R6)+          ;:CLEAR MEMORY LOCATION
          CMP      #SWR,R6        ;:DONE?
          BNE      -6            ;:LOOP BACK IF NO
          MOV      #STACK,SP      ;:SETUP THE STACK POINTER
          ;:INITIALIZE A FEW VECTORS
          MOV      #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
          MOV      #340,@IOTVEC+2 ;:LEVEL 7
          MOV      #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
          MOV      #340,@EMTVEC+2 ;:LEVEL 7
          MOV      #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
          MOV      #340,@TRAPVEC+2 ;:LEVEL 7
          MOV      $ENDCT,$EOPCT  ;:SETUP END-OF-PROGRAM COUNTER
          MOV      #176543,$HINUM ;:PRIME THE RANDOM NUMBER GENERATOR
          MOV      #123456,$LONUM ;:BOTH HIGH AND LOW WORDS
          CLR      $TIMES        ;:INITIALIZE NUMBER OF ITERATIONS
          CLR      $ESCAPE       ;:CLEAR THE ESCAPE ON ERROR ADDRESS
          MOV8     #1,$ERMAX      ;:ALLOW ONE ERROR PER TEST
          MOV      #.,$SLPADR    ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
          MOV      #.,$SLPERR    ;:SETUP THE ERROR LOOP ADDRESS
004736 012706 001114
004742 005026
004744 022706 001154
004750 001374
004752 012706 001100
004756 012737 025356 000020
004764 012737 000340 000022
004772 012737 021620 000030
005000 012737 000340 000032
005006 012737 026052 000034
005014 012737 000340 000036
005022 013737 021514 021506
005030 012737 176543 026524
005036 012737 123456 026526
005044 005037 001220
005050 005037 001222
005054 112737 000001 001131
005062 012737 005062 001122
005070 012737 005070 001124

```

```

::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
::EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
005076 013746 000004      MOV    @#ERRVEC,-(SP)    ;;SAVE ERROR VECTOR
005102 012737 005136 000004      MOV    #64$,@#ERRVEC    ;;SET UP ERROR VECTOR
005110 012737 177570 001154      MOV    #DSWR,SWR    ;;SETUP FOR A HARDWARE SWICH REGISTER
005116 012737 177570 001156      MOV    #DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
005124 022777 177777 174022      CMP    #-1,@SWR    ;;TRY TO REFERENCE HARDWARE SWR
005132 001012      BNE    66$    ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                ;;AND THE HARDWARE SWR IS NOT: - -1
005134 000403      BR    65$    ;;BRANCH IF NO TIMEOUT
005136 012716 005144      64$: MOV    #65$,(SP)    ;;SET UP FOR TRAP RETURN
005142 000002      RTI
005144 012737 000176 001154      65$: MOV    #SWREG,SWR    ;;POINT TO SOFTWARE SWR
005152 012737 000174 001156      MCV    #DISPREG,DISPLAY
005160 012637 000004      66$: MOV    (SP)+,@#ERRVEC    ;;RESTORE ERROR VECTOR

005164 005037 001242      CLR    $PASS    ;;CLEAR PASS COUNT
005170 132737 000200 001255      BITB    #APTSIZE,$ENVM    ;;TEST USER SIZE UNDER APT
005176 001403      BEQ    67$    ;;YES,USE NON-APT SWITCH
005200 012737 001256 001154      MOV    #SWREG,SWR    ;;NO,USE APT SWITCH REGISTER
005206      67$:
34      ;SETUP "TIMEOUT" TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
35 005206 012737 004576 000004      MOV    #BADTMO,ERRVEC    ;;SETUP FOR UNEXPECTED TIMEOUT
36 005214 012737 000300 000006      MOV    #PR6,ERRVEC+2    ;;LEVEL 6
37
38      .SBTTL TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005222 005227 177777      INC    #-1    ;;FIRST TIME?
005226 001034      BNE    68$    ;;BRANCH IF NO
005230 022737 021560 000042      CMP    #SENDAD,@#42    ;;ACT-11?
005236 001430      BEQ    68$    ;;BRANCH IF YES
005240 104401 005246      TYPE    ,69$    ;;TYPE ASCIZ STRING
005244 000425      BR    68$    ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ <CRLF>@CZRMVBO - RM05/3/2 EXTENDED DRIVE TEST@<CRLF>
005320      68$:
      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
005320 005737 000042      TST    @#42    ;;ARE WE RUNNING UNDER XXDP/ACT?
005324 001012      BNE    70$    ;;BRANCH IF YES
005326 123727 001254 000001      CMPB    $ENV,#1    ;;ARE WE RUNNING UNDER APT?
005334 001406      BEQ    70$    ;;BRANCH IF YES
005336 023727 001154 000176      CMP    SWR,#SWREG    ;;SOFTWARE SWITCH REG SELECTED?
005344 001005      BNE    71$    ;;BRANCH IF NO
005346 104406      GTSWR    ;;GET SOFT-SWR SETTINGS
005350 000403      BR    71$
005352 112737 000001 001150      70$: MOVB    #1,$AUTOB    ;;SET AUTO-MODE INDICATOR
005360      71$:

39
40 005360 012700 001174      MOV    #SREGAD,RO    ;;FIRST ADDRESS
41 005364 005020      1$: CLR    (RO)+    ;;CLEAR VARIABLE STORAGE
42 005366 022700 001224      CMP    #SBELL,RO    ;;DONE?
43 005372 001374      BNE    1$    ;;NO--BRANCH
44 005374 013737 001530 001164      MOV    TPS,$TPS    ;;SETUP THE STATUS AND BUFFER REG'S
45 005402 013737 001532 001166      MOV    TPB,$TPB    ;;FOR THE TYPE ROUTINE
46
47      ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
48      ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
49

```

50	005410	005037	001470		CLR	XXDP	;CLEAR 'XXDP' LOAD DEVICE STORAGE
51	005414	122737	C00016	000041	(MPB	#16,@#41	;LOADED FROM AN RM05/3/2 ?
52	005422	001160			BNE	4\$	;BR IF NOT
53	005424	013737	000040	001470	MOV	@#40,XXDP	;GET DEVICE INDICATOR AND NUMBER
54	005432	122737	000007	001470	(MPB	#7,XXDP	;IS IT A VALID NUMBER ?
55	005440	103002			BHIS	2\$	;YES
56	005442	105037	001470		CLRB	XXDP	;NO, DEFAULT TO DRIVE 0
57	005446	005737	000042	2\$:	TST	@#42	;CHAIN MODE OR ACT11 AUTO ACCEPT ?
58	005452	001425			BEQ	3\$	;BR IF NEITHER
59	005454	104401	005462		TYPE	,73\$	::TYPE ASCIZ STRING
	005460	000412			BR	72\$	::GET OVER THE ASCIZ
				::73\$:	.ASCIZ	<CRLF>/NOT TESTING DRIVE /	
				72\$:			
60	005506	005046			CLR	-(SP)	;CLEAR WORD ON STACK
61	005510	113716	001470		MOVB	XXDP,(SP)	;GET DRIVE ADDRESS
62	005514	104403			TYPOS		;TYPE THE ADDRESS
63	005516	001			.BYTE	1	;ONLY 1 CHARACTER
64	005517	000			.BYTE	0	;SUPRESS LEADING ZEROS
65	005520	104401	001231		-YPE	,SCLF	;CR-LF
66	005524	000517			BR	4\$	;GET NUMBER OF DRIVES
67							
68	005526	005227	177777	3\$:	INC	#-1	;FIRST TIME THRU HERE ?
69	005532	001114			BNE	4\$	;NO
70	005534	104401	005542		TYPE	,75\$	::TYPE ASCIZ STRING
	005540	000410			BR	74\$	::GET OVER THE ASCIZ
				::75\$:	.ASCIZ	<CRLF>/TO TEST DRIVE /	
				74\$:			
71	005562	005046			CLR	-(SP)	;CLEAR WORD ON STACK
72	005564	113716	001470		MOVB	XXDP,(SP)	;GET DRIVE ADDRESS
73	005570	104403			TYPOS		;TYPE DRIVE ADDRESS
74	005572	001			.BYTE	1	;ONLY 1 CHARACTER
75	005573	000			.BYTE	0	;SUPPRESS LEADING ZEROS
76	005574	104401	005602		TYPE	,77\$	::TYPE ASCIZ STRING
	005600	000431			BR	76\$	::GET OVER THE ASCIZ
				::77\$:	.ASCIZ	/, HALT PROGRAM, REMOVE RRD P PACK AND REPLACE IT/<CRLF>	
				76\$:			
77	005664	104401	005672		TYPE	,78\$	::TYPE ASCIZ STRING
	005670	000435			BR	4\$	::GET OVER THE ASCIZ
				::78\$:	.ASCIZ	/WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>	
				4\$:			
81	005764	004737	024060		JSR	PC,\$TKINT	;TURN ON THE TTY KEYBOARD INTERRUPT
82	005770	005227	177777		INC	#-1	;SEE IF FIRST START
83	005774	001002			BNE	SRTINT	;BR IF NOT
84	005776	004737	046624		JSR	PC,GETADR	;GET OR CHECK THE RH/RM ADDRESS
85							
86	006002	105037	001472	SRTINT:	CLRB	ERRCN	;CLEAR DRV 0 ERROR COUNT
89	006006	105037	001473		CLRB	ERRCN+1	;CLEAR DRV 1 ERROR COUNT
	006012	105037	001474		CLRB	ERRCN+2	;CLEAR DRV 2 ERROR COUNT
	006016	105037	001475		CLRB	ERRCN+3	;CLEAR DRV 3 ERROR COUNT
	006022	105037	001476		CLRB	ERRCN+4	;CLEAR DRV 4 ERROR COUNT
	006026	105037	001477		CLRB	ERRCN+5	;CLEAR DRV 5 ERROR COUNT
	006032	105037	001500		CLRB	ERRCN+6	;CLEAR DRV 6 ERROR COUNT
	006036	105037	001501		CLRB	ERRCN+7	;CLEAR DRV 7 ERROR COUNT
90	006042	004737	027364		JSR	PC,LP.AVL	;CHECK FOR A LINE PRINTER
91	006046	012737	000001	001120	MOV	#1,\$ICNT	;SET ITERATION COUNT TO 1
92	006054	004737	035450		JSR	PC,GETSWR	;GO CHECK FOR CONTROL SWITCHES
93							

```

94 006060 004737 027426          SETVEC: JSR    PC,ST.CLK      ;INITIALIZE THE CLOCK
95 006064 004737 C40660          JSR    PC,RMINIT    ;CHECK THE DRIVE STATUS
96 006070 012737 177777 040610  MOV    #-1,SAVEFG    ;SET THE SAVE REGISTERS FLAG
97 006076 004737 027334          JSR    PC,CNTCLR    ;GO CLEAR MASSBUS CONTROLLER
98 006102 005037 177776          CLR    PS            ;ENSURE THE PRIORITY = 0
99 006106 005227 177777          INC    #-1           ;FIRST TIME THRU HERE ?
100 006112 001403          BEQ    1$              ;BR IF NO
101 006114 005737 001322          TST    CNTRLC        ;CONTROL 'C' SWITCH SET ?
102 006120 001105          BNE    SRTDRV          ;CONTINUE IF YES
103 006122 005004          1$: CLR    R4            ;DRIVE TABLE POINTER
104 006124 104401 047313          TYPE   ,UNSTAT      ;TYPE 'UNIT STATUS'
105 006130 104401 001231          2$: TYPE   ,$CRLF     ;CR-LF
106 006134 010446          MOV    R4,-(SP)        ;SAVE R4 FOR TYPEOUT
          ;TYPE DRIVE NUMBER
          ;GO TYPE--OCTAL ASCII
          ;TYPE 2 DIGIT(S)
          ;SUPPRESS LEADING ZEROS
          ;TYPE 4 SPACFS
107 006142 104401 050343          TYPE   ,BLNKS4      ;CHECK DRIVE'S STATUS
108 006146 105764 040532          TSTB   DRVSTA(R4)   ;BR IF UNSAFE
109 006152 100416          BMI     5$              ;BR IF ONLINE
110 006154 001020          BNE     6$              ;SEE IF OFFLINE OR NONEXISTENT
111 006156 105764 040542          TSTB   DRVTP(R4)   ;BR IF NONEXISTENT
112 006162 001404          BEQ     3$              ;BR IF OFFLINE
113 006164 100006          BPL     ,NOTRM        ;DRIVE NOT AN RM05/3/2
114 006166 104401 047377          TYPE   ,NOTRM        ;CHECK NEXT DRIVE
115 006172 000452          BR      11$
116
117 006174 104401 047352          3$: TYPE   ,NOTPRS     ;DRIVE NOT PRESENT
118 006200 000447          BR      11$              ;CHECK NEXT DRIVE
119
120 006202 104401 047331          4$: TYPE   ,UNTOFF    ;DRIVE OFFLINE
121 006206 000416          BR      8$              ;PRINT DRIVE TYPE
122
123 006210 104401 047367          5$: TYPE   ,NOTSAF    ;DRIVE UNSAFE
124 006214 000413          BR      8$              ;PRINT DRIVE TYPE
125
126 006216 005737 001470          6$: TST    XXDP        ;LOADED FROM THIS DEVICE ?
127 006222 001406          BEQ     7$              ;BR IF NO
128 006224 123704 001470          CMPB   XXDP,R4      ;LOADED FROM THIS DRIVE ?
129 006230 001003          BNE     7$              ;BR IF NO
130 006232 104401 047420          TYPE   ,LODEV        ;DRIVE IS LOAD DEVICE
131 006236 000430          BR      11$
132 006240 104401 047342          7$: TYPE   ,UNTON        ;DRIVE ONLINE
133 006244 104401 050345          8$: TYPE   ,BLNKS2      ;TYPE 2 SPACES
134 006250 005000          CLR     R0
135 006252 116400 040542          MOVB   DRVTP(R4),R0 ;GET DRIVE TYPE
136 006256 012737 047445 006316  MOV    #SRM03,10$  ;ASSUME ADDRESS OF RM03 MESSAGE
137 006264 122700 000004          CMPB   #4,R0      ;IS DEVICE AN RM03 ?
138 006270 001411          BEQ     9$              ;TYPE IT IF YES
139 006272 012737 047440 006316  MOV    #SRM02,10$  ;ADDRESS OF RM02 MESSAGE
140 006300 122700 000005          CMPB   #5,R0      ;IS DEVICE AN RM02 ?
141 006304 001403          BEQ     9$              ;BR IF YES
142 006306 012737 047452 006316  MOV    #SRM05,10$  ;ADDRESS OF RM05 MESSAGE
143
144 006314 104401          9$: TYPE   ;TYPE THE DRIVE TYPE MESSAGE
145 006316 000000          10$: .WORD 0 ;MESSAGE ADDRESS HERE
146

```

147	006320	005204		11\$:	INC	R4	: INCREMENT DRIVE NUMBER/TABLE POINTER
148	006322	020427	000010		CMP	R4,#8.	: FINISHED ?
149	006326	001300			BNE	2\$	: BR IF NOT
150	006330	104401	001231		TYPE	,SCLF	: CR-LF

```

1          .SBTTL  GET UNIT STATUS
2
3 006334 005737 001322      SRTDRV: TST      CNTRLC      ;CONTROL 'C' START/RESTART?
4 006340 001427              BEQ      2$      ;NO--BRANCH
5 006342 013746 001320      MOV      SAVCSW,-(SP) ;GET THE PREVIOUS 'C.SWR' CONTENTS
6 006346 063716 001314      ADD      C.SWR,(SP)  ;SET UP TO SEE IF 'BIT00' IS DIFFERENT
7 006352 032726 000001      BIT      #BIT00,(SP)+ ;IS 'BIT00' DIFFERENT ?
8 006356 001405              BEQ      1$      ;BR IF NOT
9 006360 013737 001314 001320      MOV      C.SWR,SAVCSW ;STORE PRESENT 'C.SWR' VALUE
10 006366 004737 027704      JSR      PC,LODFLT ;RESET PARAMETERS TO THEIR DEFAULT VALUES
11 006372 004737 035700      JSR      PC,GT.PRM ;GET PARAMETERS
12 006376 005737 001332      TST      TSTNMS ;ANY TEST SELECTED THIS CYCLE?
13 006402 001034              BNE      6$      ;BR IF YES
14 006404 005737 001334      TST      TSTNMS+2 ;ANY TEST SELECTED THIS CYCLE ?
15 006410 001031              BNE      6$      ;BR IF YES
16 006412 104401 047727      TYPE      ,NOTEST ;TYPE 'NO TESTS SPECIFIED'
17 006416 000765              BR      1$
18 006420 004737 027704      2$:      JSR      PC,LODFLT ;SETUP DEFAULT PARAMETERS
19 006424 005037 001330      CLR      DRVSEL ;NO DRIVES SELECTED
20 006430 005000              CLR      R0 ;DETERMINE THE DRIVES THAT
21 006432 012701 000001      MOV      #1,R1 ;ARE AVAILABLE FOR TESTING
22
23 006436 105760 040532      3$:      TSTB     DRVSTA(R0) ;IS DRIVE ON-LINE ?
24 006442 003411              BLE      5$      ;BR IF NO
25 006444 005737 001470      TST      XXDP ;LOADED FROM THIS DEVICE ?
26 006450 001403              BEQ      4$      ;BR IF NO
27 006452 123700 001470      CMPB     XXDP,R0 ;LOADED FROM THIS DRIVE ?
28 006456 001403              BEQ      5$      ;BR IF YES
29 006460 156037 040636 001330 4$:      BISB     ATABIT(R0),DRVSEL ;YES, SELECT DRIVE FOR TESTING
30 006466 005200              5$:      INC      R0 ;TRY NEXT DRIVE
31 006470 106301              ASLB     R1 ;ANY MORE DRIVES TO CHECK ?
32 006472 001361              BNE      3$      ;BR IF YES
33
34 006474 005037 040612      6$:      CLR      SEEKFG ;CLEAR SEEK FLAG
35 006500 032737 000400 001314      BIT      #SW08,C.SWR ;DO SEEK BEFORE DATA TRANSFER?
36 006506 001002              BNE      7$      ;YES--BRANCH
37 006510 005137 040612      COM      SEEKFG ;NO
38 006514              7$:      TYPE      ,DRIVES ;'DRIVES(S) TO BE TESTED'
39 006514 104401 047457      CLR      $ENDCT ;DETERMINE PASSES TO MAKE AND
40 006520 005037 021514      CLR      R0 ;THE DRIVES TO BE TESTED
41 006524 005000              MOV      DRVSEL,R1 ;ANY DRIVES SELECTED?
42 006526 013701 001330      BNE      9$      ;YES--BRANCH
43 006532 001017              TYPE      ,NONE ;'NONE'
44 006534 104401 047510      TYPE      ,%CRLF ;CR-LF
45 006540 104401 001231      TST      #42 ;ANY MONITOR PRESENT ?
46 006544 005737 000042      BNE      8$      ;BR IF YES
47 006550 001002              JMP      START2 ;RETURN TO '^C' INPUT
48 006552 000137 004712      8$:      DEC      R0 ;THESE TWO LOOPS ARE ADDED TO
49 006556 005300              BNE      -2 ;WAIT FOR TTY
50 006560 001376              DEC      R0
51 006562 005300              BNE      -2
52 006564 001376              JMP      $GET42 ;RETURN CONTROL TO MONITOR
53 006566 000137 021550
54
55 006572 006201              9$:      ASR      R1 ;REPORT THE DRIVES TO BE TESTED
56 006574 103011              BCC      10$
57 006576 005237 021514      INC      $ENDCT ;GIVE THIS DRIVE A PASS

```

58	006602	010046			MOV	R0,-(SP)	::SAVE R0 FOR TYPEOUT
	006604	104403			TYPOS		::GO TYPE--OCTAL ASCII
	006606	001			.BYTE	1	::TYPE 1 DIGIT(S)
	006607	000			.BYTE	0	::SUPPRESS LEADING ZEROS
59	006610	005701			TST	R1	::MORE DRIVES?
60	006612	001404			BEQ	11\$	::NO--BRANCH
61	006614	104401	047515		TYPE	,CL MA	::
62	006620	005200		10\$:	INC	R0	::FORM DRIVE NUMBER
63	006622	000763			BR	9\$	
64							
65	006624	104401	001231		TYPE	,\$CRLF	::CR-LF
66	006630	013737	021514	021506	MOV	\$ENDCT,\$EOPCT	
67	006636	005737	001342		TST	CLKSTA	::IS KW11-P AVAILABLE ?
68	006642	003006			BGT	12\$	::BR IF YES
69	006644	032737	036000	001332	BIT	#36000,TSTNMS	::ANY TIMING TESTS TO BE PERFORMED ?
70	006652	001402			BEQ	12\$	::BR IF NO
71	006654	104401	047520		TYPE	,NOCLOK	::TYPE NO KW11-P CLOCK MESSAGE
72	006660	000414		12\$:	BR	RSTR11	

```

1      .SBTTL  PROGRAM RESTARTS HERE
2
3      .ENABL  LSB
4
5 006662 005737 001330      RSTART: TST      DRVSEL      ;ANY DRIVES SELECTED ?
6 006666 001022              BNE      3$              ;BR IF YES
7 006670 005737 000042      TST      @#42             ;ANY MONITOR PRESENT ?
8 006674 001402              BEQ      1$              ;BR IF NO
9 006676 000137 021550      JMP      $GET42           ;RETURN CONTROL TO MONITOR
10 006702 104401 047703     1$:  TYPE      ,NODRVS      ;TYPE 'NO DRIVES TO TEST'
11 006706 000137 004712      JMP      START2         ;NO--GET DRIVE ENTRY & RESTART AT BEGINNING
12
13 006712 005037 001352     RSTR1: CLR      CHKDRV      ;START WITH DRIVE 0 AGAIN
14 006716 012737 000001 001354 MOV      #1,DRVMSK
15 006724 033737 001354 001330 2$:  BIT      DRVMSK,DRVSEL ;IS THIS DRIVE SELECTED?
16 006732 001006              BNE      4$              ;YES--GO CHECK IF DRIVE IS READY FOR TESTING
17 006734 005237 001352     3$:  INC      CHKDRV      ;MOVE TO NEXT DRIVE NUMBER
18 006740 106337 001354      ASLB      DRVMSK         ;DONE TESTING ALL DRIVES ?
19 006744 103762              BCS      RSTR1          ;BR IF YES
20 006746 000766              BR       2$              ;NO--CHECK DRIVE SELECT
21
22 006750 013702 001352     4$:  MOV      CHKDRV,R2      ;PICKUP THE DRIVE NUMBER
23 006754 105762 040532      TSTB     DRVSTA(R2)      ;IS DESIRED DRIVE ON-LINE?
24 006760 003007              BGT      5$              ;YES, BRANCH
25 006762 104011              EMT      11              ;DRIVE SELECTED IS NOT ONLINE
26 006764 043737 001354 001330 BIC      DRVMSK,DRVSEL ;DESELECT DRIVE FROM TEST
27 006772 005337 021506      DEC      $EOPCT          ;ADJUST 'EOP' COUNT
28 006776 000731              BR       RSTART         ;RETURN
29
30 007000 004737 027334     5$:  JSR      PC,CNTCLR      ;GO CLEAR MASSBUS CONTROLLER
31 007004 005037 177776      CLR      PS              ;ENSURE THE PRIORITY = 0
32 007010 010237 047016      MOV      R2,DPB.A         ;SET THE DRIVE NUMBER INTO THE DPB'S
33 007014 010237 047036      MOV      R2,DPB.B
34 007020 010237 047056      MOV      R2,DPB.C
35 007024 010237 047076      MOV      R2,DTADPB
36 007030 004737 030336      JSR      PC,LDCMD          ;LOAD COMMAND INTO DPB.B AND DPB.C
37 007034 012737 021334 001350 MOV      #$EOP,BYPASS ;IF ERROR GO TO END OF PROGRAM
38 007042 112737 000020 047017 MOVB     #20,DPB.A+1 ;ASSUME 16 BIT FORMAT
39 007050 032737 000001 001314 BIT      #BIT00,C.SWR ;16 BIT FORMAT REQUESTED ?
40 007056 001402              BEQ      6$              ;BR IF YES
41 007060 105037 047017      CLRB     DPB.A+1         ;CLEAR THE 'FMT16' BIT
42 007064 112737 000143 047020 6$:  MOVB     #SETFORM,DPB.A+2 ;SET THE FORMAT BIT PER DPB.A+1
43 007072 004037 030402      JSR      R0,CALL.A         ;GO EXECUTE THE COMMAND
44 007076 112737 000107 047020 MOVB     #RECAL,DPB.A+2 ;RECAL=COMMAND
45 007104 004037 030402      JSR      R0,CALL.A         ;GO EXECUTE THE COMMAND
46 007110 104401 001231      TYPE     ,$CRLF          ;CR-LF
47 007114 104401 047625      TYPE     ,MSDRIV        ;TYPE 'DRIVE '
48 007120 010246      MOV      R2,-(SP)              ;SAVE R2 FOR TYPEOUT
   007122 104403      TYPOS     2                      ;GO TYPE--OCTAL ASCII
   007124 002              .BYTE     2                  ;TYPE 2 DIGIT(S)
   007125 000              .BYTE     0                  ;SUPPRESS LEADING ZEROS
49 007126 104401 047515      TYPE     ,COMMA          ;TYPE ','
50 007132 104401 047605      TYPE     ,SERIAL         ;TYPE 'MBA SNA'
51 007136 012700 000004      MOV      #4,R0           ;FOUR DIGITS TO TYPE
52 007142 013701 047170      MOV      RM,REG+30,R1      ;SERIAL NUMBER
53 007146 005002              CLR      R2              ;ZERO
54 007150 006101              ROL      R1              ;PUT THE NEXT DIGIT

```

```

55 007152 006102      ROL      R2      ;INTO R2
56 007154 006101      ROL      R1
57 007156 006102      ROL      R2
58 007160 006101      ROL      R1
59 007162 006102      ROL      R2
60 007164 006101      ROL      R1
61 007166 006102      ROL      R2
62 007170 062702 000060  ADD      #0,R2      ;MAKE IT ASCII
63 007174 010227      MOV      R2,(PC)+    ;SAVE IT
64 007176 000000      .WORD      0
65 007200 104401 007176 8$:      TYPE      ,8$      ;TYPE
66 007204 005300      DEC      R0      ;ALL DIGITS TYPED?
67 007206 003357      BGT      7$      ;NO -- BRANCH
68 007210 104401      TYPE      ,$CRLF    ;CR-LF
69 007214 113737 001464 001131  MOVB     ERR.CT,$ERMAX ;SETUP MAX ERROR COUNT
70
71      .DSABL      LSB

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
43
44

```

:////////////////////////
:*IN THE DESCRIPTIONS OF THE BELOW TESTS THE VARIABLES USED
:*AND THEIR DEFAULT VALUES (UNLESS SPECIFIED OTHERWISE) ARE:
:*
:*MNEMONIC      VALUE      VARIABLE
:*-----
:*R              1          ITERATIONS (REPEATS)
:*FC              0          FIRST CYLINDER ADDRESS
:*LC              822.       LAST CYLINDER ADDRESS
:*IC              1          INCREMENT VALUE
:*NC OF NC1      FC+IC      NEW OR MODIFIED CYLINDER
:*                ADDRESS
:*NC2             LC-IC      NEW OR MODIFIED CYLINDER
:*                ADDRESS
:*
:*FT              0          FIRST TRACK ADDRESS
:*LT              4 OR 18.   LAST TRACK ADDRESS
:*IT              1          INCREMENT VALUE
:*NT              FT+IT      NEW OR MODIFIED TRACK ADDRESS
:*
:*FS              0          FIRST SECTOR ADDRESS
:*LS              31.        LAST SECTOR ADDRESS
:*
:////////////////////////

```

.SBTTL SEEK TESTS

```

:////////////////////////
:*THE SEEK TESTS WILL BE EXECUTED USING IMPLIED SEEKS. THESE
:*IMPLIED SEEKS WILL BE PERFORMED BY 'READ HEADER AND DATA'
:*COMMANDS TO TRACK 'FT' SECTOR 'FS' OF THE DESIRED CYLINDER.
:*THE WORD COUNT WILL BE SET SUCH THAT ONLY THE CYLINDER AND
:*TRACK/SECTOR WORDS OF THE HEADER ARE READ.
:////////////////////////

```

```

:*****
:*TEST 0      RECAL/RANDOM SEEK TEST
:*THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE COMMAND
:*AND THEN SEEK TO A RANDOM CYLINDER BETWEEN 'FC' AND 'LC'. AT
:*THE COMPLETION OF BOTH COMMANDS, STATUS INDICATORS ARE CHECKED
:*TO ENSURE THAT NO ERRORS OCCURRED.
:*****
:STO:

```

```

007222
007222 000240
007224 033737 001540 001332
007232 001002
007234 000137 007464

007240 012737 000000 001116
007246 004737 030142
007252 012737 007372 001124
007260 013737 002336 001220
007266 112737 000031 001131
007274 012737 000000 001240

```

```

NOP
BIT BITS+<0*2>,TSTNMS ;DO THIS TEST?
BNE .+6 ;BR IF YES
JMP TST1 ;NO--JUMP TO TEST1

MOV #0,$TSTNM ;SET TEST #0 AND CLEAR ($ERFLG)
JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
MOV #TEST0,$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
MOV RPT,$TIMES ;GET THE ITERATION COUNT
MOVB #25,$ERMAX ;MAX ERRORS ALLOWED FOR TEST
MOV #0,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

```

Line	Address	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464	Op46
------	---------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

007560	013746	001240		MOV	\$TESTN,-(SP)	::SAVE \$TESTN FOR TYPEOUT
007564	104403			TYPOS		::GO TYPE--OCTAL ASCII
007566	002			.BYTE	2	::TYPE 2 DIGIT(S)
007567	000			.BYTE	0	::SUPPRESS LEADING ZEROS
81 007570	005037	001466		CLR	BASFLG	::CLEAR BAD SECTOR ENCOUNTER FOR THE DRIVE
82 007574	113737	002354	047046	MOVB	FS,DPB.B+10	::FS
83 007602	113737	002356	047066	MOVB	LS,DPB.C+10	::LS
84 007610	113737	002346	047047	MOVB	FT,DPB.B+11	::FT
85 007616	113737	002350	047067	MOVB	LT,DPB.C+11	::LT
86 007624	013737	002340	047050	MOV	FC,DPB.B+12	::FC
87 007632	013737	002342	047070	MOV	LC,DPB.C+12	::LC
92 007640	012737	007670	001350	MOV	#EXIT1,BYPASS	::GO TO EXIT1 ON ERROR
007646	012737	007654	001122	MOV	#TEST1,\$LPADR	::SETUP LOOP ADDRESS
007654				TEST1:		
93 007654	012706	001100		MOV	#STACK,SP	::SET THE STACK POINTER
94 007660	004037	030766		JSR	RO,CALL.C	::GO EXECUTE THE COMMAND
95 007664	004037	030550		JSR	RO,CALL.B	::GO EXECUTE THE COMMAND
96 007670	000004			EXIT1:	SCOPE	::CALL SCOPE ROUTINE

 *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.

007672				TST2:		
007672	000240			NOP		
007674	033737	001544	001332	BIT	BITS+<2*2>,\$TSTNMS	::DO THIS TEST?
007702	001002			BNE	+6	::BR IF YES
007704	000137	010134		JMP	TST3	::NO--JUMP TO TEST3
007710	012737	000002	001116	MOV	#2,\$TSTNM	::SET TEST #2 AND CLEAR (\$ERFLG)
007716	004737	030142		JSR	PC,LODPRM	::LOAD THE PARAMETERS FOR THE TEST
007722	012737	010026	001124	MOV	#TEST2,\$LPERR	::SETUP THE LOOP ON ERROR ADDRESS
007730	013737	002336	001220	MOV	RPT,\$TIMES	::GET THE ITERATION COUNT
007736	112737	000031	001131	MOVB	#25,\$ERMAX	::MAX ERRORS ALLOWED FOR TEST
007744	012737	000002	001240	MOV	#2,\$TESTN	::SET TEST NUMBER IN APT MAIL BOX
110 007752	032777	010000	171174	BIT	#SW12,\$SWR	::INHIBIT TYPING TEST NUMBER ?
007760	001406			BEO	+16	::BR IF YES
007762	104401	047617		TYPE	,MSGTST	::TYPE 'TEST'
007766	013746	001240		MOV	\$TESTN,-(SP)	::SAVE \$TESTN FOR TYPEOUT
007772	104403			TYPOS		::GO TYPE--OCTAL ASCII
007774	002			.BYTE	2	::TYPE 2 DIGIT(S)
007775	000			.BYTE	0	::SUPPRESS LEADING ZEROS
111 007776	012737	010004	001122	MOV	#1,\$,\$LPADR	::SETUP LOOP ADDRESS
112 010004	113737	002354	047046	MOVB	FS,DPB.B+10	::FS
113 010012	113737	002346	047047	MOVB	FT,DPB.B+11	::FT
117 010020	012737	010132	001350	MOV	#EXIT2,BYPASS	::GO TO EXIT2 ON ERROR

010026
 118 010026 013737 002340 047050
 119 010034 012737 010034 001124
 010042 012706 001100
 120 010046
 121 010046 004037 030550
 122 010052 063737 002344 047050
 123 010060 023737 002342 047050
 124 010066 002367
 125 010070 013737 002342 047050
 126 010076 012737 010076 001124
 010104 012706 001100
 127 010110
 128 010110 004037 030550
 129 010114 163737 002344 047050
 130 010122 023737 002340 047050
 131 010130 003767
 132 010132 000004
 133
 140
 141

TEST2:
 MOV FC,DPB.B+12 ;FC
 MOV #,\$LPERR ;SETUP THE ERROR LOOP ADDRESS
 MOV #STACK,SP ;LOAD THE STACK POINTER
 INCSK:
 JSR R0,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 #,\$LPERR ;SETUP THE ERROR LOOP ADDRESS
 MOV #STACK,SP ;LOAD THE STACK POINTER
 DECSK:
 JSR R0,CALL.B ;GO EXECUTE THE COMMAND
 SUB IC,DPB.B+12
 CMP FC,DPB.B+12
 BLE DECSK
 EXIT2: SCOPE ;CALL SCOPE ROUTINE

 *TEST 3 STEPPING SEEK TEST
 *THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0, 1, 2, 4,
 *8, 16, 32, 64, 128, 256 AND 512. AT THE COMPLETION OF EACH
 *SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE
 *PROPER OPERATION.

010134
 010134 000240
 010136 033737 001546 001332
 010144 001002
 010146 000137 010354
 010152 012737 000003 001116
 010160 004737 030142
 010164 012737 010270 001124
 010172 013737 002336 001220
 010200 112737 000031 001131
 010206 012737 000003 001240
 142 010214 032777 010000 170732
 010222 001406
 010224 104401 047617
 010230 013746 001240
 010234 104403
 010236 002
 010237 000

TEST3:
 NOP
 BIT BITS+<3*2>,TSTNMS ;DO THIS TEST?
 BNE .+6 ;BR IF YES
 JMP TST4 ;NO--JUMP TO TEST4
 MOV #3,\$TSTNM ;SET TEST #3 AND CLEAR (\$ERFLG)
 JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
 MOV #TEST3,\$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
 MOV RPT,\$TIMES ;GET THE ITERATION COUNT
 MOV #25,\$ERMAX ;MAX ERRORS ALLOWED FOR TEST
 MOV #3,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
 BIT #SW12,\$SWR ;INHIBIT TYPING TEST NUMBER ?
 BEQ .+16 ;BR IF YES
 TYPE ,MSGTST ;TYPE 'TEST'
 MOV \$TESTN,-(SP) ;SAVE \$TESTN FOR TYPEOUT
 TYP0S ;GO TYPE--OCTAL ASCII
 .BYTE 2 ;TYPE 2 DIGIT(S)
 .BYTE 0 ;SUPPRESS LEADING ZEROS

143 010240 012737 010246 001122
 144 010246 113737 002354 047046
 145 010254 113737 002346 047047
 149 010262 012737 010352 001350
 010270
 150 010270 013737 002340 047050
 151 010276 012737 010276 001124
 010304 012706 001100
 152 010310 004037 030550

1\$:
 MOV #1\$, \$LPADR ;SETUP TEST LOOP ADDRESS
 MOV FS,DPB.B+10 ;FS
 MOV FT,DPB.B+11 ;FT
 MOV #EXIT3,BYPASS ;GO TO BYPASS ON ERROR
 TEST3:
 MOV FC,DPB.B+12 ;FC
 MOV #,\$LPERR ;SETUP THE ERROR LOOP ADDRESS
 MOV #STACK,SP ;LOAD THE STACK POINTER
 JSR R0,CALL.B ;GO EXECUTE THE COMMAND

```

153 010314 013701 002344      MOV      IC,R1          ;CYLINDER 1
154 010320 012737 010320 001124  MOV      #,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
      010326 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
155 010332 010137 047050      MOV      R1,DPB.B+12      ;DESIRED CYLINDER
156 010336 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
157 010342 006301          ASL      R1          ;MOVE TO NEXT CYLINDER
158 010344 020137 002342      CMP      R1,LC          ;DONE?
159 010350 003770          BLE      1$          ;NO--LOOP
160 010352 000004      EXIT3:  SCOPE      ;CALL SCOPE ROUTINE
161
169
170

```

```

*****
*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.
*****

```

```

171 010354 000240          TST4:  NOP
      010356 033737 001550 001332  BIT      BITS+<4*2>,TSTNMS      ;DO THIS TEST?
      010364 001002          BNE      +6          ;BR IF YES
      010366 000137 010766          JMP      TST5          ;NO--JUMP TO TEST5

      010372 012737 000004 001116  MOV      #4,$TSTNM      ;SET TEST #4 AND CLEAR ($ERFLG)
      010400 004737 030142          JSR      PC,LODPRM      ;LOAD THE PARMETERS FOR THE TEST
      010404 012737 010524 001124  MOV      #TEST4,$LPERR      ;SETUP THE LOOP ON ERROR ADDRESS
      010412 013737 002336 001220  MOV      RPT,$TIMES      ;GET THE ITERATION COUNT
      010420 112737 000031 001131  MOV      #25,$ERMAX      ;MAX ERRORS ALLOWED FOR TEST
      010426 012737 000004 001240  MOV      #4,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX

      010434 032777 010000 170512  BIT      #SW12,$SWR      ;INHIBIT TYPING TEST NUMBER ?
      010442 001406          BEQ      +16          ;BR IF YES
      010444 104401 047617          TYPE      $MSGTST      ;TYPE 'TEST'
      010450 013746 001240          MOV      $TESTN,-(SP)      ;SAVE $TESTN FOR TYPEOUT
      010454 104403          TYPOS      ;GO TYPE--OCTAL ASCII
      010456 002          .BYTE      2          ;TYPE 2 DIGIT(S)
      010457 000          .BYTE      0          ;SUPPRESS LEADING ZEROS

172 010460 012737 010466 001122      MOV      #1$,$LPADR      ;SETUP LOOP ADDRESS
173 010466 113737 002354 047046 1$:  MOV      FS,DPB.B+10      ;FS
174 010474 113737 002346 047047      MOV      FT,DPB.B+11      ;FT
182 010502 012737 010764 001350      MOV      #EXIT4,BYPASS      ;GO TO EXIT4 ON ERROR
      010510 005002          CLR      R2          ;CLEAR STALL SWITCH (NO STALL)
      010512 032737 010000 001314  BIT      #SW12,C.SWR      ;STALL REQUIRED?
      010520 001401          BEQ      TEST4          ;NO--BRANCH
      010522 005102          COM      R2          ;YES--SET SWITCH
      010524          TEST4:

183 010524 013701 002340          MOV      FC,R1          ;SET NC TO FC
184 010530 005037 001436          CLR      STALL0      ;START AT ZERO IF STALLS REQUIRED
185 010534 012737 010534 001124  MOV      #,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
      010542 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
186 010546 010137 047050      MOV      R1,DPB.B+12      ;NC
187 010552 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
188 010556 005702          IST      R2          ;STALL?
189 010560 001403          BEQ      2$          ;NO--BRANCH

```

```

190 010562 004037 032326 JSR RO,STALL ;YES-GO TO STALL ROUTINE
191 010566 001436 .WORD STALLO ;TIME POINTER
192 010570 013737 002340 047050 2$: MOV FC,DPB.B+12 ;FC
193 010576 004037 030550 JSR RO,CALL.B ;GO EXECUTE THE COMMAND
194 010602 005702 TST R2 ;STALL?
195 010604 001413 BEQ 3$ ;NO--BRANCH
196 010606 004037 032326 JSR RO,STALL ;YES--GO TO STALL ROUTINE
197 010612 001436 .WORD STALLO ;TIME POINTER
198 010614 005237 001436 INC STALLO ;UPDATE THE TIME
199 010620 023737 001462 001436 CMP MXSTAL,STALLO ;TIME TO BIG?
200 010626 003347 BGT 1$ ;NO--BRANCH
201 010630 005037 001436 CLR STALLO ;YES--START OVER AT ZERO
202 010634 063701 002344 3$: ADD IC,R1 ;MOVE TO NEXT CYLINDER
203 010640 020137 002342 CMP R1,LC ;LAST CYLINDER COMPLETED?
204 010644 003740 BLE 1$ ;NO--BRANCH
205 010646 013701 002342 MOV LC,R1 ;SET NC TO LC
206 010652 012737 010652 001124 MOV #,$LPERR ;SETUP THE ERROR LOOP ADDRESS
207 010660 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
208 010664 010137 047050 4$: MOV R1,DPB.B+12 ;NC
209 010670 004037 030550 JSR RO,CALL.B ;GO EXECUTE THE COMMAND
210 010674 005702 TST R2 ;STALL?
211 010676 001403 BEQ 5$ ;NO--BRANCH
212 010700 004037 032326 JSR RO,STALL ;YES--GO TO STALL ROUTINE
213 010704 001436 .WORD STALLO ;TIME POINTER
214 010706 013737 002342 047050 5$: MOV LC,DPB.B+12 ;LC
215 010714 004037 030550 JSR RO,CALL.B ;GO EXECUTE THE COMMAND
216 010720 005702 TST R2 ;STALL?
217 010722 001413 BEQ 6$ ;NO--BRANCH
218 010724 004037 032326 JSR RO,STALL ;YES--GO TO STALL ROUTINE
219 010730 001436 .WORD STALLO ;TIME POINTER
220 010732 005237 001436 INC STALLO ;UPDATE STALL TIME
221 010736 023737 001462 001436 CMP MXSTAL,STALLO ;TIME TOO BIG?
222 010744 003347 BGT 4$ ;NO--BRANCH
223 010746 005037 001436 CLR STALLO ;YES--SET STALL TIME BACK TO ZERO
224 010752 163701 002344 6$: SUB IC,R1 ;NEXT CYLINDER
225 010756 020137 002340 CMP R1,FC ;DONE?
226 010762 002340 BGE 4$ ;NO--BRANCH
227 010764 000004 EXIT4: SCOPE ;CALL SCOPE ROUTINE
228
229

```

```

*****
*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.
*****

```

```

010766
010766 000240
010770 033737 001552 001332 NOP
010776 001002 BIT BITS+<5*2>,TSTNMS ;DO THIS TEST?
011000 000137 011212 BNE +6 ;BR IF YES
JMP TST6 ;NO--JUMP TO TEST6

```

```

240 011004 012737 000005 001116      MOV    #5,$TSTNM      ;SET TEST #5 AND CLEAR ($ERFLG)
    011012 004737 030142              JSR    PC,LODPRM      ;LOAD THE PARMETERS FOR THE TEST
    011016 012737 011122 001124      MOV    #TEST5,$LPERR    ;SETUP THE LOOP ON ERROR ADDRESS
    011024 013737 002336 001220      MOV    RPT,$TIMES      ;GET THE ITERATION COUNT
    011032 112737 000031 001131      MOV    #25,$ERMAX      ;MAX ERRORS ALLOWED FOR TEST
    011040 012737 000005 001240      MOV    #5,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX

    011046 032777 010000 170100      BIT    #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
    011054 001406              BEQ    .+16              ;BR IF YES
    011056 104401 047617              TYPE   ,MSGTST          ;TYPE 'TEST'
    011062 013746 001240              MOV    $TESTN,-(SP)      ;SAVE $TESTN FOR TYPEOUT
    011066 104403              TYPPOS   ;GO TYPE--OCTAL ASCII
    011070 002              .BYTE    2              ;TYPE 2 DIGIT(S)
    011071 000              .BYTE    0              ;SUPPRESS LEADING ZEROS
  
```

```

241 011072 012737 011100 001122      MOV    #1$,$LPADR      ;SETUP LOOP ADDRESS
242 011100 113737 002354 047046 1$:  MOV    FS,DPB.B+10      ;FS
243 011106 113737 002346 047047      MOV    FT,DPB.B+11      ;FT
247 011114 012737 011210 001350      MOV    #EXIT5,BYPASS   ;GO TO EXIT5 ON ERROR

    011122              TEST5:  MOV    FC,R1              ;START NC1 AT FC
248 011122 013701 002340              MOV    LC,R2              ;START NC2 AT LC
249 011126 013702 002342              MOV    #,$LPERR          ;SETUP THE ERROR LOOP ADDRESS
250 011132 012737 011132 001124      MOV    #STACK,SP       ;LOAD THE STACK POINTER
    011140 012706 001100              1$:  MOV    R1,DPB.B+12      ;NC1
251 011144 010137 047050              JSR    R0,CALL.B        ;GO EXECUTE THE COMMAND
252 011150 004037 030550              MOV    R2,DPB.B+12      ;NC2
253 011154 010237 047050              JSR    R0,CALL.B        ;GO EXECUTE THE COMMAND
254 011160 004037 030550              ADD    IC,R1              ;NEXT NC1
255 011164 063701 002344              SUB    IC,R2              ;NEXT NC2
256 011170 163702 002344              CMP    R1,LC              ;DONE?
257 011174 020137 002342              BGT    EXIT5            ;YES--BRANCH
260 011200 003003              CMP    R2,FC            ;NO--BRANCH
261 011202 020237 002340              BGE    1$              ;CALL SCOPE ROUTINE
262 011206 002356
263 011210 000004
264
273
274
  
```

EXIT5: SCOPE

;CALL SCOPE ROUTINE

```

*****
*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
*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:
  
```

```

011212
011212 000240
011214 033737 001554 001332
011222 001002
011224 000137 011502
  
```

```

NOP
BIT    BITS+<6*2>,$TSTNMS      ;DO THIS TEST?
BNE    .+6                      ;BR IF YES
JMP    TST7                    ;NO--JUMP TO TEST7
  
```

```

011230 012737 000006 001116
011236 004737 030142
011242 012737 011346 001124
011250 013737 002336 001220
011256 112737 000031 001131
  
```

```

MOV    #6,$TSTNM      ;SET TEST #6 AND CLEAR ($ERFLG)
JSR    PC,LODPRM      ;LOAD THE PARMETERS FOR THE TEST
MOV    #TEST6,$LPERR    ;SETUP THE LOOP ON ERROR ADDRESS
MOV    RPT,$TIMES      ;GET THE ITERATION COUNT
MOVB    #25,$ERMAX      ;MAX ERRORS ALLOWED FOR TEST
  
```

```

275 011264 012737 000006 001240      MOV      #6,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
      011272 032777 010000 167654      BIT      #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
      011300 001406                      BEQ      .+16          ;BR IF YES
      011302 104401 047617                      TYPE    ,MSGTST      ;TYPE 'TEST'
      011306 013746 001240      MOV      $TESTN,-(SP)      ;;SAVE $TESTN FOR TYPEOUT
      011312 104403                      TYPOS    ;GO TYPE--OCTAL ASCII
      011314 002                      .BYTE     2          ;TYPE 2 DIGIT(S)
      011315 000                      .BYTE     0          ;SUPPRESS LEADING ZEROS

276 011316 012737 011324 001122      MOV      #1$, $LPADR      ;SETUP LOOP ADDRESS
277 011324 113737 002354 047046 1$:  MOV      FS,DPB.B+10      ;FS
278 011332 113737 002346 047047      MOV      FT,DPB.B+11      ;FT
282 011340 012737 011500 001350      MOV      #EXIT6,BYPASS    ;GO TO EXIT6 ON ERROR
      011346                      TEST6:
283 011346 013701 002340      MOV      FC,R1          ;PICKUP 'FC'
284 011352 013702 002342      MOV      LC,R2          ;FORM LAST CYLINDER THAT
285 011356 162702 000005                      SUB      #5,R2          ;IS AVAILABLE FOR TESTING
286 011362 012737 011362 001124      MOV      #,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
      011370 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
287 011374 020102                      1$:  CMP      R1,R2          ;LAST CYLINDER
290 011376 003040      BGT      EXIT6          ;YES--BRANCH
291 011400 010137 047050      MOV      R1,DPB.B+12      ;NC
292 011404 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
293 011410 062737 000004 047050      ADD      #4,DPB.B+12      ;NC+4
294 011416 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
295 011422 162737 000003 047050      SUB      #3,DPB.B+12      ;NC+1
296 011430 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
297 011434 062737 000002 047050      ADD      #2,DPB.B+12      ;NC+3
298 011442 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
299 011446 162737 000001 047050      SUB      #1,DPB.B+12      ;NC+2
300 011454 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
301 011460 062737 000003 047050      ADD      #3,DPB.B+12      ;NC+5
302 011466 004037 030550      JSR      R0,CALL.B      ;GO EXECUTE THE COMMAND
303 011472 063701 002344      ADD      IC,R1
304 011476 000736      BR      1$
305 011500 000004      EXIT6:  SCOPE          ;CALL SCOPE ROUTINE
306
315
316

```

```

*****
;TEST 7      RANDOM SEEK TEST
;THIS TEST PERFORMS RANDOM SEEK OPERATIONS BETWEEN CYLINDERS 'FC'
;AND '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 PARAMTERS 'FT' AND 'LT'.
*****
TST7:

```

```

011502 000240      NOP
011504 033737 001556 001332      BIT      BITS+<7*2>,$TSTNMS      ;DO THIS TEST?
011512 001002      BNE      .+6          ;BR IF YES
011514 000137 012100      JMP      TST10          ;NO--JUMP TO TEST10

011520 012737 000007 001116      MOV      #7,$TSTNM      ;SET TEST #7 AND CLEAR ($ERFLG)
011526 004737 030142      JSR      PC,LODPRM      ;LOAD THE PARMETERS FOR THE TEST
011532 012737 011642 001124      MOV      #TEST7,$LPERR      ;SETUP THE LOOP ON ERROR ADDRESS

```

```

317 011540 013737 002336 001220      MOV      RPT,$TIMES      ;GET THE ITERATION COUNT
    011546 112737 000031 001131      MOV      #25,$SERMAX    ;MAX ERRORS ALLOWED FOR TEST
    011554 012737 000007 001240      MOV      #7,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX

    011562 032777 010000 167364      BIT      #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
    011570 001406                      BEQ      .+16             ;BR IF YES
    011572 104401 047617                      TYPE    ,MSGTST      ;TYPE 'TEST'
    011576 013746 001240      MOV      $TESTN,-(SP)      ;SAVE $TESTN FOR TYPEOUT
    011602 104403                      TYP0S    ;GO TYPE--OCTAL ASCII
    011604 002                          .BYTE    2        ;TYPE 2 DIGIT(S)
    011605 000                          .BYTE    0        ;SUPPRESS LEADING ZEROS

318 011606 113737 002346 047047      MOV      FT,DPB.B+11    ;LOAD STARTING TRACK ADDRESS
319 011614 112737 000105 047020      MOV      #SEEK,DPB.A+2  ;SEEK=COMMAND
320 011622 013704 040650      MOV      RMADR,R4      ;UNIBUS ADDRESS OF THE RH/RM
325 011626 012737 012076 001350      MOV      #EXIT7,BYPASS  ;ERROR TERMINATION ADDRESS
    011634 012737 011642 001122      MOV      #TEST7,$LPADR  ;SETUP THE LOOP ON TEST ADDRESS
    011642                                TEST7:
    011642 012706 001100      MOV      #STACK,SP      ;SETUP THE STACK POINTER
327 011646 013737 002340 047050      MOV      FC,DPB.B+12    ;INITIAL CYLINDER ADDRESS
328 011654 023737 002340 002342      CMP      FC,LC        ;CYLINDER LIMITS THE SAME ?
329 011662 001422                      BEQ      1$           ;BR IF THEY ARE
330 011664 004737 026426      JSR      PC,$RAND      ;CYCLE THE RANDOM NUMBER GENERATOR
331 011670 013746 026524      MOV      $HNUM,-(SP)    ;USE THE HIGH RANDOM NUMBER
332 011674 005046                      CLR      -(SP)         ;UPPER DIVIDEND
333 011676 013746 002342      MOV      LC,-(SP)      ;FORM THE DIVISOR
334 011702 005216                      INC      (SP)         ;INCREMENT
335 011704 163716 002340      SUB      FC,(SP)      ;SUBTRACT THE LOWER LIMIT
336 011710 004737 026530      JSR      PC,$DIV      ;DIVIDE
337 011714 062637 047050      ADD      (SP)+,DPB.B+12  ;ADD THE REMAINDER TO THE INITIAL CYLINDER
338 011720 005726                      TST      (SP)+        ;DISCARD THE QUOTIENT
339 011722 013737 047050 047030      MOV      DPB.B+12,DPB.A+12 ;COPY NEW CYLINDER ADDRESS
340 011730                                1$:
    011730 012737 011730 001124      MOV      #,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
    011736 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
341 011742 004037 030402      JSR      R0,CALL.A      ;GO EXECUTE THE COMMAND
342 011746 012737 011746 001124      MOV      #,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
    011754 012706 001100      MOV      #STACK,SP      ;LOAD THE STACK POINTER
343 011760 113764 047016 000010      MOV      DPB.A,RMCS2(R4) ;SELECT THE DRIVE
344 011766 016446 000020      MOV      RMLA(R4),-(SP) ;GET THE LOOK AHEAD REGISTER
345 011772 006316                      ASL      (SP)         ;ALIGN THE SECTOR ADDRESS
346 011774 006316                      ASL      (SP)         ;ALIGN THE SECTOR ADDRESS
347 011776 000316                      SWAB    (SP)         ;PUT ADDRESS IN LOWER BYTE
348 012000 105766 000001      TSTB    1(SP)        ;IN THE 1ST 20% OF SECTOR ?
349 012004 001401                      BEQ      2$           ;BR IF YES
350 012006 105216                      INCB    (SP)         ;INCREMENT THE SECTOR ADDRESS
351 012010 105216                      INCB    (SP)         ;INCREMENT THE SECTOR ADDRESS
352 012012 112637 047106      MOV      (SP)+,DTADPB+10 ;LOAD THE DPB
353 012016 013746 002470      MOV      PRMLMT+24,-(SP) ;PUT LAST SECTOR ADDRESS ON THE STACK
354 012022 005216                      INC      (SP)         ;INCREMENT IT
355 012024 122637 047106      CMPB    (SP)+,DTADPB+10 ;NEW SECTOR ADDRESS TOO LARGE ?
356 012030 103007                      BHS     4$           ;BR IF NOT
357 012032 103403                      BLO     3$           ;BR IF ADDRESS IS 2 GREATER
358 012034 105037 047106      CLRB    DTADPB+10      ;RESET TO SECTOR ADDRESS 0
359 012040 000403                      BR      4$           ;CONTINUE
360 012042 112737 000001 047106      MOV      #1,DTADPB+10 ;RESET ADDRESS TO SECTOR 1
361 012050                                4$:
  
```

	012050	004037	030550		JSR	RO,CALL.B	:GO EXECUTE THE COMMAND
362	012054	105237	047047		INCB	DPB.B+11	:INCREMENT THE TRACK ADDRESS
363	012060	123737	047047	002350	CMPB	DPB.B+11,LT	:MAXIMUM ?
366	012066	101403			BLOS	EXIT7	:BR IF NOT
367	012070	113737	00.346	047047	MOVB	FT,DPB.B+11	:RELOAD STARTING TRACK ADDRESS
368	012076	000004			EXIT7:	SCOPE	:CALL SCOPE ROUTINE

 369
 391
 392

```

:*****
: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 (RMLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING INTO
:POSITION.  THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND
:FOR THAT SECTOR.
: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 IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW
:HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS.  THE NECESSARY
:TIME DEPENDENT PARAMETERS OCCUR FREQUENTLY ENOUGH WITHIN THE REQUIRED
:RANGE TO PERMIT THIS TEST TO BE EFFECTIVE.
:*****

```

	012100				TST10:		
	012100	000240			NOP		
	012102	033737	001560	001332	BIT	BITS+<10*2>,TSTNMS	:DO THIS TEST?
	012110	001002			BNE	.+6	:BR IF YES
	012112	000137	013254		JMP	TST11	:NO--JUMP TO TEST11
	012116	012737	000010	001116	MOV	#10,\$TSTNM	:SET TEST #10 AND CLEAR (\$ERFLG)
	012124	004737	030142		JSR	PC,LODPRM	:LOAD THE PARMETERS FOR THE TEST
	012130	012737	012356	001124	MOV	#TEST10,\$LPERR	:SETUP THE LOOP ON ERROR ADDRESS
	012136	013737	002336	001220	MOV	RPT,\$TIMES	:GET THE ITERATION COUNT
	012144	112737	000031	001131	MOVB	#25,\$SERMAX	:MAX ERRORS ALLOWED FOR TEST
	012152	012737	000010	001240	MOV	#10,\$TESTN	:SET TEST NUMBER IN APT MAIL BOX
393	012160	032777	010000	166766	BIT	#SW12,@SWR	:INHIBIT TYPING TEST NUMBER ?
	012166	001406			BEQ	.+16	:BR IF YES
	012170	104401	047617		TYPE	,MSGTST	:TYPE 'TEST'
	012174	013746	001240		MOV	\$TESTN,-(SP)	:SAVE \$TESTN FOR TYPEOUT
	012200	104403			TYPOS		:GO TYPE--OCTAL ASCII
	012202	003			.BYTE	3	:TYPE 3 DIGIT(S)
	012203	000			.BYTE	0	:SUPPRESS LEADING ZEROS
394	012204	012737	012212	001122	MOV	#1\$,\$LPADR	:SETUP THE LOOP ADDRESS
395	012212				1\$:		
	012212	112737	000105	047020	MOVB	#SEEK,DPB.A+2	:SEEK=COMMAND
396	012220	112737	000161	047100	MOVB	#WRITE,DTADPB+2	:COMMAND
397	012226	113737	002346	047107	MOVB	FT,DTADPB+11	:TRACK ADDRESS FOR THE WRITE
398	012234	013737	002340	047030	MOV	FC,DPB.A+12	:CYLINDER ADDRESS FOR THE SEEK

```

399 012242 013737 002340 047110      MOV      FC,DTADPB+12      ;CYLINDER ADDRESS FOR THE WRITE
400 012250 013737 002340 002370      MOV      FC,NC1          ;STARTING CYLINDER
401 012256 013737 002342 001450      MOV      LC,DELTA        ;GET LAST CYLINDER
402 012264 163737 002340 001450      SUB      FC,DELTA        ;CALCULATE DELTA COUNT
403 012272 023737 001450 002344      CMP      DELTA,IC        ;IS CALCULATED COUNT <= 'IC' ?
404 012300 003403                BLE      2$              ;BR IF YES
405 012302 013737 002344 001450      MOV      IC,DELTA        ;CYLINDER INCREMENT VALUE
406 012310 012737 176000 047102      MOV      #-<256.*4.>,DTADPB+4 ;WORD COUNT
407 012316 012737 054522 047104      MOV      #BUFFER,DTADPB+6 ;BUFFER ADDRESS
408 012324 005000                CLR      R0              ;PATTERN POINTER (WC PATTERN)
409 012326 004737 034404      JSR      PC,SETBUF      ;LOAD THE WRITE BUFFER
410 012332 005001                CLR      R1              ;CLEAR REGISTER
411 012334 113701 047016      MOV      DPB.A,R1        ;LOAD DRIVE ADDRESS
412 012340 013704 040650      MOV      RMADR,R4        ;UNIBUS ADDRESS OF THE RH/RM
413 012344 004737 046336      JSR      PC,CLRQUE      ;CLEAR THE OPERATION QUEUES
454 012350 012737 013252 001350      MOV      #EXIT10,BYPASS ;ERROR EXIT FROM TEST
                                TEST10:
012356 012737 012356 001124      MOV      #.,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
012364 012706 001100                MOV      #STACK,SP        ;LOAD THE STACK POINTER
012370 012737 000340 177776      MOV      #PR7,PS        ;SET PRIORITY TO 7
012376 005737 001342                TST      CLKSTA        ;SEE WHICH CLOCK ON SYSTEM
012402 001415                BEQ      3$              ;BR IF NO CLOCK
012404 100405                BMI      1$              ;BR IF KW11-L CLOCK
012406 017746 167076      MOV      @PKV,-(SP)      ;SAVE THE VECTOR
012412 013746 001510      MOV      PKV,-(SP)      ;SAVE THE VECTOR ADDRESS
012416 000404                BR      2$              ;CONTINUE
012420 017746 167076      MOV      @LKV,-(SP)      ;SAVE THE 'L' CLOCK VECTOR
012424 013746 001522      MOV      LKV,-(SP)      ;SAVE THE VECTOR ADDRESS
012430 012776 013206 000000      MOV      #TST10B,@(SP) ;CHANGE THE VECTOR
012436 012777 032734 026206      MOV      #DORTI,@RMVEC ;CHANGE THE RM VECTOR
012444 012737 000010 001444      MOV      #8.,SEKTR      ;LOAD THE SEEK TIMER
012452 012764 000040 000010      MOV      #CLR,RMCS2(R4) ;INIT THE MASSBUS
012460 110164 000010      MOV      R1,RMCS2(R4) ;RESELECT THE DRIVE
012464 013764 047030 000034      MOV      DPB.A+12,RMDC(R4) ;LOAD THE CYLINDER ADDRESS
012472 013737 047030 001366      MOV      DPB.A+12,CYL.DS ;CYLINDER ADDRESS FOR ERROR MESSAGE
012500 112764 000105 000000      MOV      #SEEK,RMCS1(R4) ;START THE SEEK
012506 005037 177776      CLR      PS              ;CLEAR THE PRIORITY
012512 105764 000012      TST      RMDS(R4)      ;HAS THE DRIVE FINISHED ?
012516 100402                BMI      5$              ;BR IF IT HAS
012520 000001                WAIT             ;WAIT FOR THE OPERATION TO COMPLETE
012522 000773                BR      4$              ;CONTINUE
012524 012737 000340 177776      MOV      #PR7,PS        ;CHANGE PRIORITY TO MAX
012532 032764 040000 000012      BIT      #ERR,RMDS(R4) ;ERROR ?
012540 001412                BEQ      6$              ;BR IF NOT
012542 012702 047016      MOV      #DPB.A,R2        ;DPB POINTER
012546 004737 045654      JSR      PC,SVRH70      ;SAVE THE REGISTERS
012552 104023                EMT      23          ;ERROR DURING SEEK
012554 012764 000040 000010      MOV      #CLR,RMCS2(R4) ;INIT THE MASSBUS
012562 110164 000010      MOV      R1,RMCS2(R4) ;RESELECT THE DRIVE
012566 012777 043306 026056      MOV      #ISR,@RMVEC    ;SETUP THE RM VECTOR
012574 005737 001342      TST      CLKSTA        ;WHICH CLOCK
012600 001405                BEQ      TST10A      ;BR IF NONE
012602 016676 000002 000000      MOV      2(SP),@(SP)    ;RELOAD THE CLOCK VECTOR
012610 062706 000004      ADD      #4,SP        ;CORRECT THE STACK POINTER
                                TST10A:
455 012614 012737 012614 001124      MOV      #.,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
012622 012706 001100                MOV      #STACK,SP        ;LOAD THE STACK POINTER

```

456	012626	110164	000010		MOVB	R1, RMCS2(R4)	; SELECT THE DRIVE
457	012632	016446	000020		MOV	RMLA(R4), -(SP)	; GET THE LOOK AHEAD REGISTER
458	012636	006316			ASL	(SP)	; ALIGN THE SECTOR ADDRESS
459	012640	006316			ASL	(SP)	; ALIGN THE SECTOR ADDRESS
460	012642	000316			SWAB	(SP)	; PUT ADDRESS IN LOWER BYTE
461	012644	122766	000300	000001	CMPB	#300, 1(SP)	; IN THE LAST 20% OR SECTOR ?
462	012652	001001			BNE	2\$	; BR IF NOT
463	012654	105216			INCB	(SP)	; INCREMENT THE SECTOR ADDRESS
464	012656	105216			INCB	(SP)	; INCREMENT THE SECTOR ADDRESS
465	012660	112637	047106		MOVB	(SP)+, DTADPB+10	; LOAD THE DPB
466	012664	013746	002470		MOV	PRMLT+24, -(SP)	; PUT MAXIMUM SECTOR ADDRESS ON THE STACK
467	012670	005216			INC	(SP)	; INCREMENT PAST THE MAXIMUM ADDRESS
468	012672	122637	047106		CMPB	(SP)+, DTADPB+10	; NEW SECTOR ADDRESS TOO LARGE ?
469	012676	101007			BHI	4\$	; BR IF NOT
470	012700	103403			BLO	3\$	; BR IF ADDRESS IS 2 GREATER THAN MAXIMUM
471	012702	105037	047106		CLRB	DTADPB+10	; RESET TO SECTOR ADDRESS 0
472	012706	000403			BR	4\$	; CONTINUE
473	012710	112737	000001	047106	MOVB	#1, DTADPB+10	; RESET ADDRESS TO SECTOR 1
474	012716	012703	047102		MOV	#DTADPB+4, R3	; POINTER
475	012722	012764	000111	000000	MOV	#DRVCLR, RMCS1(R4)	; CLEAR THE DRIVE
476	012730	012364	000002		MOV	(R3)+, RMWC(R4)	; LOAD THE WORD COUNT
477	012734	012364	000004		MOV	(R3)+, RMBA(R4)	; LOAD THE BUFFER ADDRESS
478	012740	012364	000006		MOV	(R3)+, RMDA(R4)	; LOAD THE TRACK/SECTOR ADDR
479	012744	005037	047114		CLR	DTADPB+16	; RESET 'DONE' INDICATOR
480	012750	012737	047076	040572	MOV	#DTADPB, TRNSWT	; LOAD 'TRANSFER' DPB ADDRESS
481	012756	010137	040634		MOV	R1, DTUW	; ADDRESS OF DRIVE TRANSFERRING
482	012762	112761	000001	040522	MOVB	#1, DRVACT(R1)	; SET DRIVE ACTIVE INDICATOR
483	012770	006301			ASL	R1	; SHIFT DRIVE ADDRESS
484	012772	012761	001750	040614	MOV	#1000., TIMER(R1)	; SETUP THE OPERATION TIMER
485	013000	006201			ASR	R1	; RESTORE R1
486	013002	013764	047100	000000	MOV	DTADPB+2, RMCS1(R4)	; START THE OPERATION
487	013010	005037	177776		CLR	PS	; CLEAR THE PRIORITY
488	013014	004037	031224		JSR	R0, DRVCL1	; WAIT FOR OPERATION TO COMPLETE
489	013020	023727	001446	001750	CMP	SEKCNT, #1000.	; FINISHED SEEKS ?
490	013026	001026			BNE	6\$	; BR IF NOT
491	013030	005037	001446		CLR	SEKCNT	; CLEAR THE SEEK COUNT
492	013034	063737	002344	002370	ADD	IC, NC1	; ADD THE INCREMENT
493	013042	023737	002370	002342	CMP	NC1, LC	; EXCEEDED THE CYLINDER LIMIT ?
528	013050	103100			BHIS	EXIT10	; BR IF IT HAS
	013052	013737	002342	001450	MOV	LC, DELTA	; GET THE NEXT 'ZONE' ADDRESS
	013060	163737	002370	001450	SUB	NC1, DELTA	; CHECK THE DIFFERENCE
	013066	023737	002344	001450	CMP	IC, DELTA	; DIFFERENCE GREATER THAN THE INCREMENT ?
	013074	101003			BHI	6\$	; BR IF IT IS
	013076	013737	002344	001450	MOV	IC, DELTA	; USE THE INCREMENT PARAMETER
	013104	005237	001446		INC	SEKCNT	; COUNT THE NEXT SEEK
	013110	023737	002340	002342	CMP	FC, LC	; BEGINNING AND ENDING CYLINDERS THE SAME ?
	013116	001002			BNE	7\$	; BR IF NOT
	013120	000137	012356		JMP	TEST10	; BR IF THEY ARE
	013124	013737	002370	047030	MOV	NC1, DPB.A+12	; RESET THE CYLINDER ADDRESS
	013132	004737	026426		JSR	PC, \$RAND	; CYCLE THE RANDOM NUMBER GENERATOR
	013136	013746	026524		MOV	\$HINUM, -(SP)	; USE THE HIGH RANDOM NUMBER
	013142	005046			CLR	-(SP)	; CLEAR THE UPPER DIVIDEND
	013144	013746	001450		MOV	DELTA, -(SP)	; FORM THE DIVISOR
	013150	005216			INC	(SP)	; INCREMENT
	013152	004737	026530		JSR	PC, \$DIV	; DIVIDE
	013156	062637	047030		ADD	(SP)+, DPB.A+12	; ADD THE REMAINDER TO THE INITIAL CYLINDER
	013162	005726			TST	(SP)+	; DISCARD THE QUOTIENT

013164	023737	047030	047110	CMP	DPB.A+12,DTADPB+12	:SAME CYLINDER SELECTED AS LAST TIME ?
013172	001754			BEQ	7\$	:BR IF IT WAS
013174	013737	047030	047110	MOV	DPB.A+12,DTADPB+12	:COPY NEW CYLINDER ADDRESS
013202	000137	012356		JMP	TEST10	:CONTINUE
013206				TST10B:		
013206	005337	001444		DEC	SEKTMR	:DECREMENT THE SEEK TIMER
013212	001016			BNE	1\$	:CONTINUE IF NOT DONE
013214	012702	047016		MOV	#DPB.A,R2	:DPB ADDRESS
013220	004737	045654		JSR	PC,SVRH70	:SAVE THE REGISTERS
013224	004024			EMT	24	:TIMEOUT DURING SEEK
013226	012764	000040	000010	MOV	#CLR,RMCS2(R4)	:INIT THE MASSBUS
013234	110164	000010		MOV	R1,RMCS2(R4)	:RESELECT THE DRIVE
013240	016676	000002	000000	MOV	2(SP),@ (SP)	:RESTORE THE CLOCK VECTOR ADDRESS
013246	000401			BR	EXIT10	:ABORT THE TEST
529 013250	000002			1\$: RTI		:RETURN
530 013252	000004			EXIT10:	SCOPE	:CALL SCOPE ROUTINE

531
542
543

 *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:

013254	000240			NOP		
013254	033737	001562	001332	BIT	BITS+<11*2>,TSTNMS	:DO THIS TEST?
013264	001002			BNE	.+6	:BR IF YES
013266	000137	013520		JMP	TST12	:NO--JUMP TO TEST12
013272	012737	000011	001116	MOV	#11,\$TSTNM	:SET TEST #11 AND CLEAR (SERFLG)
013300	004737	030142		JSR	PC,LODPRM	:LOAD THE PARAMETERS FOR THE TEST
013304	012737	013440	001124	MOV	#TEST11,\$LPERR	:SETUP THE LOOP ON ERROR ADDRESS
013312	013737	002336	001220	MOV	RPT,\$TIMES	:GET THE ITERATION COUNT
013320	112737	000031	001131	MOV	#25,\$SERMAX	:MAX ERRORS ALLOWED FOR TEST
013326	012737	000011	001240	MOV	#11,\$TESTN	:SET TEST NUMBER IN APT MAIL BOX
544 013334	032777	010000	165612	BIT	#SW12,@SWR	:INHIBIT TYPING TEST NUMBER ?
013342	001406			BEQ	.+16	:BR IF YES
013344	104401	047617		TYPE	,MSGTST	:TYPE 'TEST'
013350	013746	001240		MOV	\$TESTN,-(SP)	:SAVE \$TESTN FOR TYPEOUT
013354	104403			TYPOS		:GO TYPE--OCTAL ASCII
013356	003			.BYTE	3	:TYPE 3 DIGIT(S)
013357	000			.BYTE	0	:SUPPRESS LEADING ZEROS
545 013360	012737	013366	001122	MOV	#1\$, \$LPADR	:SETUP THE LOOP ADDRESS
546 013366	113737	002354	047046	1\$: MOV	FS,DPB.B+10	:SECTOR ADDRESS
547 013374	113737	002354	047066	MOV	FS,DPB.C+10	:SECTOR ADDRESS
548 013402	113737	002346	047047	MOV	FT,DPB.B+11	:TRACK ADDRESS
549 013410	113737	002346	047067	MOV	FT,DPB.C+11	:TRACK ADDRESS
550 013416	013737	002340	047050	MOV	FC,DPB.B+12	:STARTING CYLINDER ADDRESS
551 013424	013737	002340	047070	MOV	FC,DPB.C+12	:STARTING CYLINDER ADDRESS

555	013432	012737	013516	001350	MOV	#EXIT11,BYPASS	;TEST ABORT EXIT
	013440				TEST11:		
556	013440	012706	001100		MOV	#STACK,SP	;SETUP THE STACK POINTER
557	013444				1\$:		
	013444	004037	030766		JSR	RO,CALL.C	;GO EXECUTE THE COMMAND
558	013450	004037	030550		JSR	RO,CALL.B	;GO EXECUTE THE COMMAND
559	013454	063737	002344	047070	ADD	IC,DPB.C+12	;INCREMENT THE ENDING CYLINDER ADDRESS
560	013462	023737	002342	047070	CMP	LC,DPB.C+12	;CHECK IF EXCEEDING MAXIMUM
561	013470	002365			BGE	1\$	;BR IF NOT
562	013472	013737	002340	047070	MOV	FC,DPB.C+12	;RESET ENDING CYLINDER ADDRESS
563	013500	063737	002344	047050	ADD	IC,DPB.B+12	;INCREMENT THE STARTING ADDRESS
564	013506	023737	002342	047050	CMP	LC,DPB.B+12	;EXCEEDING MAXIMUM ?
565	013514	002353			BGE	1\$	;BR IF NOT
566	013516	000004			EXIT11:	SCOPE	;CALL SCOPE ROUTINE

1
2
3
4
5
6
7
8
9
10
11
12
28
29

[illegible]

```

*****
: *TEST 12          ROTATIONAL SPEED TIMING TEST
: *THIS TEST WILL START A SEARCH TO CYLINDER 'FC', TRACK 'FT',
: *SECTOR 'FS'. 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:
:
:
: RM05/3:
:         16.67 MS/REV + OR - 2% IF 60HZ
:         16.67 MS/REV + OR - 2.5% IF 50HZ.
:
:
: RM02:
:         25.00 MS/REV + OR - 2% IF 60HZ
:         25.00 MS/REV + OR - 2.5% IF 50HZ.
:
*****

```

İŞY 12:

	013520			
	013520	000240		
	013522	033737	001564	001332
	013530	001002		
	013532	000137	014442	
	013536	012737	000012	001116
	013544	004737	030142	
	013550	012737	014116	001124
	013556	013737	002336	001220
	013564	112737	000031	001131
	013572	012737	000012	001240
30	013600	032777	010000	165346
	013606	001406		
	013610	104401	047617	
	013614	013746	001240	
	013620	104403		
	013622	003		
	013623	000		
31	013624	005737	001342	
32	013630	003002		
35	013632	000137	014440	
36	013636	012737	013636	001122
37	013644	004037	032552	
38	013650	000402		
60	013652	000137	014440	

```

NOP
BIT      BITS+<12+2> ,TSTNMS      ;DO THIS TEST?
BNE      .+6                       ;BR IF YES
JMP      TST13                     ;NO--JUMP TO TEST13

MOV      #12,$TSTNM                ;SET TEST #12 AND CLEAR ($ERFLG)
JSR      PC,LODPRM                 ;LOAD THE PARMETERS FOR THE TEST
MOV      #TEST12,$LPERR            ;SETUP THE LOOP ON ERROR ADDRESS
MOV      RPT,$TIMES                ;GET THE ITERATION COUNT
MOV      #25,$ERMAX                ;MAX ERRORS ALLOWED FOR TEST
MOV      #12,$TESTN                ;:SET TEST NUMBER IN APT MAIL BOX

BIT      #SW12,@SWR                ;INHIBIT TYPING TEST NUMBER ?
BEQ      .+16                      ;BR IF YES
TYPE     ,MSGTST                   ;TYPE 'TEST'
MOV      $TESTN,-(SP)              ;:SAVE $TESTN FOR TYPEOUT
TYPOS    ;:GO TYPE--OCTAL ASCII
.BYTE    3                         ;:TYPE 3 DIGIT(S)
.BYTE    0                         ;:SUPPRESS LEADING ZEROS

TST      CLKSTA                    ;KW11-P CLOCK?
BGT      1$                        ;YES--START TEST
JMP      EXIT12                    ;NO--JUMP TO EXIT12
MOV      #.,$LPADR                 ;SETUP LOOP ADDRESS
JSR      RO,$SRCHOO                ;DO A MASSBUS INIT & RECAL
BR       2$                        ;RETURN HERE IF NO ERROR
JMP      EXIT12                    ;RETURN HERE IF ERROR

```

```

013656 012737 013656 001122 2$: MOV #,$LPADR ;ERROR LOOP ADDRESS
013664 112737 000105 047020 MOVB #SEEK,DPB.A+2 ;SEEK=COMMAND
013672 005037 047026 CLR DPB.A+10 ;USE TRACK 0 & SECTOR 0
013676 013737 002340 047030 MOV FC,DPB.A+12 ;STARTING CYLINDER
013704 012737 014440 001350 MOV #EXIT12,BYPASS ;GO TO EXIT12 IF ERROR
013712 004037 030402 JSR R0,CALL.A ;GO EXECUTE THE COMMAND
013716 013764 002340 000034 MOV FC,RMDCC(R4) ;FC
013724 013746 002354 MOV FS,-(SP) ;FS
013730 113766 002346 000001 MOVB FT,1(SP) ;FT
013736 012664 000006 MOV (SP)+,RMDA(R4) ;LOAD FT/FS
013742 012737 014440 001222 MOV #EXIT12,$ESCAPE ;ESCAPE TO EXIT12 ON ERROR
013750 005005 CLR R5 ;COUNT UP

61 ;SETUP PARAMETER TABLE FOR RM05/RM03/RM02
62 013752 010046 MOV R0,-(SP) ;SAVE R0
63 013754 113700 047076 MOVB DTADPB,R0 ;DRIVE ADDRESS
64 013760 032737 000100 001314 BIT #SW06,C.SWR ;CHECK CONTROL SWR FOR 60 HZ.
65 013766 001425 BEQ 3$ ;BR IF YES
66 013770 012703 001650 MOV #T7B1,R3 ;LOAD 50 HZ. TABLE FOR RM02
67 013774 012737 001650 014430 MOV #T7B1,TP50 ;
68 014002 012737 001732 014436 MOV #SP7B1,TP550 ;
69 014010 122760 000005 040542 CMPB #5,DRVTYPE(R0) ;AN RM02 DRIVE ?
70 014016 001435 BEQ 4$ ;BRANCH IF SO
71 014020 012703 001630 MOV #T7B,R3 ;LOAD 50 HZ. TABLE FOR RM05/3
72 014024 012737 001630 014430 MOV #T7B,TP50 ;
73 014032 012737 001716 014436 MOV #SP7B,TP550 ;
74 014040 000424 BR 4$ ;EXIT
75 014042 012737 001640 014412 3$: MOV #T7A1,TP60 ;LOAD 60 HZ. TABLE FOR RM02
76 014050 012737 001724 014420 MOV #SP7A1,TP560 ;
77 014056 012703 001640 MOV #T7A1,R3 ;
78 014062 122760 000005 040542 CMPB #5,DRVTYPE(R0) ;AN RM02 DRIVE ?
79 014070 001410 BEQ 4$ ;BRANCH IF SO
80 014072 012737 001620 014412 MOV #T7A,TP60 ;LOAD 60 HZ. TABLE FOR RM05/3
81 014100 012703 001620 MOV #T7A,R3 ;
82 014104 012737 001710 014420 MOV #SP7A,TP560 ;
83 014112 012600 4$: MOV (SP)+,R0 ;RESTORE R0
84 014114 000240 NOP ;EXIT
87 014116 TEST12:
88 014116 012706 001100 MOV #STACK,SP ;SETUP STACK
89 014122 012701 000012 MOV #10,R1 ;TIME 10 SEARCHES
90 014126 004737 032736 JSR PC,STRTMR ;INITIALIZE THE TIMERS
91 014132 012777 014320 165350 MOV #7,$@PKV ;SETUP VECTOR IN CASE OF OVERFLOW
92 014140 012777 032734 024504 MOV #DORT1,@RMVEC ;SETUP RM VECTOR
93 014146 005077 165344 1$: CLR @PKB ;START COUNTING AT ZERO
94 014152 012777 000131 165334 MOV #131,@PKCS ;INT.EN., COUNT UP AT 100KHZ
95 014160 012714 000131 MOV #SEARCH,(R4) ;START A SEARCH
96 014164 000001 WAIT ;WAIT ON INTERRUPT
97 014166 042777 000101 165320 BIC #101,@PKCS ;STOP THE CLOCK
98 014174 032764 040000 000012 BIT #ERR,RMDS(R4) ;ERROR?
99 014202 001411 BEQ 2$ ;NO--BRANCH
100 014204 104412 SAVREG ;SAVE R0-R5
014206 012702 047076 MOV #DTADPB,R2 ;DPB POINTER
014212 004737 045654 JSR PC,SVRH70 ;SAVE ALL THE RH/RM REGISTERS
014216 004737 027334 JSR PC,CNTCLR ;GO CLEAR MASSBUS CONTROLLER
014222 104413 RESREG ;RESTORE R0-R5
101 014224 104017 EMT 17 ;DISK ERROR OCCURRED
102 014226 005077 165264 2$: CLR @PKB ;START THE COUNT AT ZERO

```

103	014232	012714	000131		MOV	#SEARCH,(R4)	:START A SEARCH
104	014236	012777	000131	165250	MOV	#131,@PKCS	:START THE CLOCK
105	014244	000001			WAIT		:WAIT ON INTERRUPT
106	014246	042777	000101	165240	BIC	#101,@PKCS	:STOP THE CLOCK
107	014254	032764	040000	000012	BIT	#ERR,RMDS(R4)	:IS 'ERR=1'?
108	014262	001411			BEQ	3\$	:NO--BRANCH
109	014264	104412			SAVREG		:SAVE R0-R5
	014266	012702	047076		MOV	#DTADPB,R2	:DPB POINTER
	014272	004737	045654		JSR	PC,SVRH70	:SAVE ALL THE RH/RM REGISTERS
	014276	004737	027334		JSR	PC,CNTCLR	:GO CLEAR MASSBUS CONTROLLER
	014302	104413			RESREG		:RESTORE R0-R5
110	014304	104017			EMT	17	:DISK ERROR OCCURRED
111	014306	004737	033000		JSR	PC,COUNT	:UPDATE THE COUNT
112	014312	005301			DEC	R1	:DONE?
113	014314	003314			BGT	1\$	:NO--BRANCH
114	014316	000420			BR	8\$	:YES--GO TO THE EXIT
115	014320	042777	000101	165166	BIC	#101,@PKCS	:STOP THE CLOCK
116	014326	005037	177776		CLR	PS	:DROP THE PRIORITY
117	014332	012600			MOV	(SP)+,R0	:PC OF WAIT+2
118	014334	005726			TST	(SP)+	:POP THE PS FROM THE STACK
119	014336	104412			SAVREG		:SAVE R0-R5
	014340	012702	047076		MOV	#DTADPB,R2	:DPB POINTER
	014344	004737	045654		JSR	PC,SVRH70	:SAVE ALL THE RH/RM REGISTERS
	014350	004737	027334		JSR	PC,CNTCLR	:GO CLEAR MASSBUS CONTROLLER
	014354	104413			RESREG		:RESTORE R0-R5
120	014356	104020			EMT	20	:CLOCK OVERFLOWED
121	014360	004737	027334		JSR	PC,CNTCLR	:GO CLEAR MASSBUS CONTROLLER
122	014364	004737	027426		JSR	PC,ST.CLK	:INITIALIZE THE CLOCK
123	014370	012777	043306	024254	MOV	#ISR,@RMVEC	:RESTORE RH/RM INT. VECTOR
124	014376	032737	000100	001314	BIT	#SW06,C.SWR	:60 HZ?
138	014404	001007			BNE	EXIT.A	:NO--BRANCH
	014406	004037	033240		JSR	R0,ITYPTIM	:TYPE THE TIMING
	014412	001620			TP60:	T7A	:TABLE ADDRESS
	014414	004037	033132		JSR	R0,SPTYP	:TYPE THE SPEC
	014420	001710			TPS60:	SP7A	
	014422	000406			BR	EXIT12	:EXIT
139	014424	004037	033240		EXIT.A: JSR	R0,ITYPTIM	:TYPE THE TIMING
140	014430	001630			TP50:	T7B	:TABLE ADDRESS
141	014432	004037	033132		JSR	R0,SPTYP	
142	014436	001716			TPS50:	SP7B	
143	014440	000004			EXIT12: SCOPE		:CALL SCOPE ROUTINE

144
153
154

 :*TEST 13 ONE CYLINDER SEEK TIMING TEST
 :*THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
 :*CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
 :*CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'. THE
 :*TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT
 :*EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK.
 :*MAXIMUM TIME IS 6.0 MS.
 :*****
 :*TEST13:

014442				NOP		
014442	000240			BIT	BITS+<13*2>,TS'NMS	:DO THIS TEST?
014444	033737	001566	001332	BNE	+6	:BR IF YES
014452	001002			JMP	TEST14	:NO--JUMP TO TEST14
014454	000137	015144				

155	014460	012737	000013	001116	MOV	#13,\$STSTNM	;SET TEST #13 AND CLEAR (\$ERFLG)
	014466	004737	030142		JSR	PC,LODPRM	;LOAD THE PARAMETERS FOR THE TEST
	014472	012737	014652	001124	MOV	#TEST13,\$LPERR	;SETUP THE LOOP ON ERROR ADDRESS
	014500	013737	002336	001220	MOV	RPT,\$TIMES	;GET THE ITERATION COUNT
	014506	112737	000031	001131	MOVB	#25,\$SERMAX	;MAX ERRORS ALLOWED FOR TEST
	014514	012737	000013	001240	MOV	#13,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
	014522	032777	010000	164424	BIT	#SW12,@SWR	;INHIBIT TYPING TEST NUMBER ?
	014530	001406			BEQ	+.16	;BR IF YES
	014532	104401	047617		TYPE	,MSGTST	;TYPE 'TEST'
	014536	013746	001240		MOV	\$TESTN,-(SP)	;SAVE \$TESTN FOR TYPEOUT
	014542	104403			TYPOS		;GO TYPE--OCTAL ASCII
	014544	003			.BYTE	3	;TYPE 3 DIGIT(S)
	014545	000			.BYTE	0	;SUPPRESS LEADING ZEROS
156	014546	005737	001342		TST	CLKSTA	;KW11-P CLOCK?
157	014552	003002			BGT	1\$	;YES--START TEST
160	014554	000137	015142		JMP	EXIT13	;NO--JUMP TO EXIT13
161	014560	012737	014560	001122	1\$: MOV	#,\$LPADR	;SETUP THE LOOP ADDRESS
162	014566	004037	032552		JSR	RO,SRCH00	;DO A MASSBUS INIT. AND RECAL
163	014572	000402			BR	2\$	;NO ERROR RETURN
175	014574	000137	015142		JMP	EXIT13	;ERROR RETURN--SCOPE LOOP CALL
	014600	012737	014600	001122	2\$: MOV	#,\$LPADR	;ERROR LOOP ADDRESS
	014606	112737	000105	047020	MOVB	#SEEK,DPB.A+2	;SEEK=COMMAND
	014614	005037	047026		CLR	DPB.A+10	;USE TRACK 0 & SECTOR 0
	014620	013737	002340	047030	MOV	FC,DPB.A+12	;STARTING CYLINDER
	014626	012737	015142	001350	MOV	#EXIT13,BYPASS	;GO TO EXIT13 IF ERROR
	014634	004037	030402		JSR	RO,CALL.A	;GO EXECUTE THE COMMAND
	014640	012703	001660		MOV	#T10,R3	;PARAMETER POINTER
	014644	012737	015142	001222	MOV	#EXIT13,\$ESCAPE	;ESCAPE TO EXIT13 ON ERROR
	014652				TEST13:		
176	014652	012706	001100		MOV	#STACK,SP	;SETUP STACK
177	014656	013737	002340	047110	MOV	FC,DTADPB+12	;START WITH BEGINNING CYLINDER
178	014664	005237	047110		INC	DTADPB+12	;INCREMENT THE BEGINNING CYLINDER
179	014670	005005			CLR	R5	;SET THE UP/DOWN SWITCH TO UP
180	014672	004737	032736		JSR	PC,STRIMR	;INITIALIZE THE TIMERS
181	014676	012777	015050	164604	MOV	#7\$,@PKV	;SETUP INCASE OF OVERFLOW
182	014704	012777	032734	023740	MOV	#DORT1,@RMVEC	;SET RM VECTOR
183	014712	005077	164600		1\$: CLR	@PKB	;START THE COUNTER AT ZERO
184	014716	013764	047110	000034	MOV	DTADPB+12,RMD(R4)	;LOAD DESIRED CYLINDER
185	014724	012714	000105		MOV	#SEEK,(R4)	;START A SEEK
186	014730	012777	000131	164556	MOV	#131,@PKCS	;START THE CLOCK
187	014736	000001			WAIT		;WAIT ON INTERRUPT
188	014740	042777	000101	164546	BIC	#101,@PKCS	;STOP THE CLOCK
189	014746	032764	040000	000012	BIT	#ERR,RMD(R4)	;ANY DISK ERRORS?
190	014754	001411			BEQ	2\$	;NO--BRANCH
191	014756	104412			SAVREG		;SAVE R0-R5
	014760	012702	047076		MOV	#DTADPB,R2	;DPB POINTER
	014764	004737	045654		JSR	PC,SVRH70	;SAVE ALL THE RH/RM REGISTERS
	014770	004737	027334		JSR	PC,CNTCLR	;GO CLEAR MASSBUS CONTROLLER
	014774	104413			RESREG		;RESTORE R0-R5
192	014776	104017			EMT	17	;DISK ERROR OCCURRED
193	015000	004737	033000		2\$: JSR	PC,COUNT	;COUNT THIS SEEKS TIME
194	015004	005705			TST	R5	;UP OR DOWN?
195	015006	001011			BNE	4\$	;DOWN--BRANCH
196	015010	005237	047110		3\$: INC	DTADPB+12	;MOVE TO NEXT CYLINDER

```

197 015014 023737 047110 002342      CMP      DTADPB+12,LC      ;OUT OF CYLINDERS?
198 015022 002733                    BLT       1$              ;NO--GO DO THE NEXT SEEK
199 015024 012705 177777              MOV      #-1,R5         ;SET UP/DOWN SWITCH TO DOWN
200 015030 000730                    BR        1$              ;GO DO THE NEXT SEEK
201 015032 005337 047110              DEC      DTADPB+12        ;MOVE TO NEXT CYLINDER
202 015036 023737 047110 002340      CMP      DTADPB+12,FC      ;OUT OF CYLINDERS?
203 015044 003322                    BGT       1$              ;NO--GO DO THE NEXT SEEK
204 015046 000420                    BR        8$              ;GO TO THE EXIT
205 015050 042777 000101 164436      BIC      #101,@PKCS      ;STOP THE CLOCK
206 015056 005037 177776              CLR      PS              ;DROP THE PRIORITY
207 015062 012600                    MOV      (SP)+,R0        ;PC OF WAIT+2
208 015064 005726                    TST      (SP)+          ;POP THE PS FROM THE STACK
209 015066 104412                    SAVREG                    ;SAVE R0-R5
    015070 012702 047076              MOV      #DTADPB,R2        ;DPB POINTER
    015074 004737 045654              JSR      PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
    015100 004737 027334              JSR      PC,CNTCLR      ;GO CLEAR MASSBUS CONTROLLER
    015104 104413                    RESREG                    ;RESTORE R0-R5
210 015106 104020                    EMT       20              ;REPORT CLOCK OVERFLOW
211 015110 004737 027334              JSR      PC,CNTCLR      ;GO CLEAR MASSBUS CONTROLLER
212 015114 004737 027426              JSR      PC,ST.CLK      ;INITIALIZE THE CLOCK
213 015120 012777 043306 023524      MOV      #ISR,@RMVEC    ;RESTORE RH/RM INT. VECTOR
214 015126 004037 033240              JSR      R0,TYPTIM      ;GO TYPE THE TIMES
    015132 001660                    TIO                      ;POINTER
215 015134 004037 033132              JSR      R0,SPTYP
216 015140 001740                    SP10
217 015142 000004                    EXIT13: SCOPE          ;CALL SCOPE ROUTINE
218
232
233
  
```

```

*****
*TEST 14      AVERAGE SEEK TIMING TEST
*THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO
*CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
*CYLINDER 'FC'. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY
*ARE WITHIN THE TOLERANCE ALLOWED FOR THE AVERAGE SEEK TIME.
*THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF 256 SEEKS).
*MAXIMUM TIME IS 30.0 MS.
*
* THERE ARE NO SPECIFICATIONS GIVEN FOR AN AVERAGE SEEK TIME
* ON THIS PARTICULAR DRIVE. THEREFORE, THIS TEST SHOULD BE
* USED FOR REFERENCE ONLY.
*
*****
  
```

```

015144
015144 000240
015146 033737 001570 001332      NOP
015154 001002                    BIT      BITS+<14*2>,TSTNMS      ;DO THIS TEST?
015156 000137 015704              BNE      .+6              ;BR IF YES
                                JMP      TST15              ;NO--JUMP TO TEST15

015162 012737 000014 001116      MOV      #14,$TSTNM      ;SET TEST #14 AND CLEAR ($ERFLG)
015170 004737 030142              JSR      PC,LODPRM      ;LOAD THE PARMETERS FOR THE TEST
015174 012737 015354 001124      MOV      #TEST14,$LPERR    ;SETUP THE LOOP ON ERROR ADDRESS
015202 013737 002336 001220      MOV      RPT,$TIMES      ;GET THE ITERATION COUNT
015210 112737 000031 001131      MOV      #25,$SERMAX    ;MAX ERRORS ALLOWED FOR TEST
015216 012737 000014 001240      MOV      #14,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX

234 015224 032777 010000 163722      BIT      #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
015232 001406                    BEQ      .+16              ;BR IF YES
  
```

015234	104401	047617				TYPE	MSGTST	:TYPE 'TEST'
015240	013746	001240				MOV	\$TESTN,-(SP)	::SAVE \$TESTN FOR TYPEOUT
015244	104403					TYPOS		::GO TYPE--OCTAL ASCII
015246	003					.BYTE	3	::TYPE 3 DIGIT(S)
015247	000					.BYTE	0	::SUPPRESS LEADING ZEROS
235	015250	005737	001342			TST	CLKSTA	:KW11-P CLOCK?
236	015254	003002				BGT	1\$	:YES--START TEST
239	015256	000137	015702			JMP	EXIT14	:NO--JUMP TO EXIT14
240	015262	012737	015262	001122	1\$:	MOV	#,\$LPADR	:SET THE LOOP ADDRESS
241	015270	004037	032552			JSR	RO,SRCH00	:DO A MASSBUS INIT & RECAL
242	015274	000402				BR	2\$	:RETURN HERE IF NO ERROR
254	015276	000137	015702			JMP	EXIT14	:RETURN HERE ON ERROR
	015302	012737	015302	001122	2\$:	MOV	#,\$LPADR	:ERROR LOOP ADDRESS
	015310	112737	000105	047020		MOVB	#SEEK,DPB.A+2	:SEEK=COMMAND
	015316	005037	047026			CLR	DPB.A+10	:USE TRACK 0 & SECTOR 0
	015322	013737	002340	047030		MOV	FC,DPB.A+12	:STARTING CYLINDER
	015330	012737	015702	001350		MOV	#EXIT14,BYPASS	:GO TO EXIT14 IF ERROR
	015336	004037	030402			JSR	RO,CALL.A	:GO EXECUTE THE COMMAND
	015342	012703	001670			MOV	#T11,R3	:PARAMETER POINTER
	015346	012737	015702	001222		MOV	#EXIT14,\$ESCAPE	::ESCAPE TO EXIT14 ON ERROR
	015354							
255	015354	012706	001100			MOV	#STACK,SP	:SETUP STACK
256	015360	012701	000200			MOV	#128,R1	:REPEAT "'FC'-'LC'-'FC'" 128 TIMES
257	015364	004737	032736			JSR	PC,STRTMR	:INIT. THE COUNTERS
258	015370	012777	015610	164112		MOV	#7\$,@PKV	:SET UP VECTOR IN CASE OF OVERFLOW
259	015376	012777	032734	023246		MOV	#DORT1,@RMVEC	:SETUP RM VECTOR
260	015404	005077	164106		1\$:	CLR	@PKB	:START COUNT AT ZERO
261	015410	013764	002342	000034		MOV	LC,RMDC(R4)	: 'MIDDLE' CYLINDER
262	015416	012764	000105	000000		MOV	#SEEK,RMCS1(R4)	:START A SEEK
263	015424	012777	000131	164062		MOV	#131,@PKCS	:START THE CLOCK
264	015432	000001				WAIT		:WAIT ON INTERRUPT
265	015434	042777	000101	164052		BIC	#101,@PKCS	:STOP CLOCK
266	015442	032764	040000	000012		BIT	#ERR,RMDS(R4)	:ERR=1?
267	015450	001411				BEQ	2\$	:NO--BRANCH
268	015452	104412				SAVREG		:SAVE R0-R5
	015454	012702	047076			MOV	#DTADPB,R2	:DPB POINTER
	015460	004737	045654			JSR	PC,SVRH70	:SAVE ALL THE RH/RM REGISTERS
	015464	004737	027334			JSR	PC,CNTCLR	:GO CLEAR MASSBUS CONTROLLER
	015470	104413				RESREG		:RESTORE R0-R5
269	015472	104017				EMT	17	:DISK ERROR OCCURRED
270	015474	005005			2\$:	CLR	R5	:SET UP/DOWN SWITCH TO UP
271	015476	004737	033000			JSR	PC,COUNT	:UPDATE THE COUNT
272	015502	005077	164010			CLR	@PKB	:START THE COUNT AT ZERO
273	015506	013764	002340	000034		MOV	FC,RMDC(R4)	:BEGINNING CYLINDER
274	015514	012764	000105	000000		MOV	#SEEK,RMCS1(R4)	:START A SEEK
275	015522	012777	000131	163764		MOV	#131,@PKCS	:START THE CLOCK
276	015530	000001				WAIT		:WAIT ON INTERRUPT
277	015532	042777	000101	163754		BIC	#101,@PKCS	:STOP THE CLOCK
278	015540	032764	040000	000012		BIT	#ERR,RMDS(R4)	:ERR=1?
279	015546	001411				BEQ	3\$	:NO--BRANCH
280	015550	104412				SAVREG		:SAVE R0-R5
	015552	012702	047076			MOV	#DTADPB,R2	:DPB POINTER
	015556	004737	045654			JSR	PC,SVRH70	:SAVE ALL THE RH/RM REGISTERS
	015562	004737	027334			JSR	PC,CNTCLR	:GO CLEAR MASSBUS CONTROLLER
	015566	104413				RESREG		:RESTORE R0-R5
281	015570	104017				EMT	17	:DISK ERROR OCCURRED

282	015572	012705	177777	3\$:	MOV	#-1,R5	;SET UP/DOWN SWITCH TO DOWN
283	015576	004737	033000		JSR	PC,COUNT	;UPDATE THE COUNT
284	015602	005301			DEC	R1	;DONE?
285	015604	003277			BGT	1\$	;NO--BRANCH
286	015606	000420			BR	8\$	;YES--EXIT
287	015610	042777	000101	163676	7\$:	BIC	#101,@PKCS
288	015616	005037	177776		CLR	PS	;STOP THE CLOCK
289	015622	012600			MOV	(SP)+,R0	;DROP THE PRIORITY
290	015624	005726			TST	(SP)+	;PC OF WAIT+2
291	015626	104412			SAVREG		;POP THE PS FROM THE STACK
	015630	012702	047076		MOV	#DTADPB,R2	;SAVE R0-R5
	015634	004737	045654		JSR	PC,SVRH70	;DPB POINTER
	015640	004737	027334		JSR	PC,CNTCLR	;SAVE ALL THE RH/RM REGISTERS
	015644	104413			RESREG		;GO CLEAR MASSBUS CONTROLLER
292	015646	104020			EMT	20	;RESTORE R0-R5
293	015650	004737	027334		8\$:	JSR	PC,CNTCLR
294	015654	004737	027426		JSR	PC,ST.CLK	;CLOCK OVERFLOWED
295	015660	012777	043306	022764	MOV	#ISR,@RMVEC	;GO CLEAR MASSBUS CONTROLLER
296	015666	004037	033240		JSR	R0,TYPTIM	;INITIALIZE THE CLOCK
	015672	001670			T11		;RESTORE RH/RM INT. VECTOR
297	015674	004037	033132		JSR	R0,SPTYP	;GO TYPE THE TIMES
298	015700	001746			SP11		;POINTER
299	015702	000004			EXIT14:	SCOPE	;CALL SCOPE ROUTINE

300
317
318

 *TEST 15 MAXIMUM SEEK TIMING TEST
 *THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO
 *CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
 *CYLINDER 'FC'. BOTH SEEKS ARE TIMED, BUT ONLY THE FORWARD SEEKS
 *ARE CHECKED TO ENSURE THEY ARE WITHIN THE TOLERANCE ALLOWED FOR
 *THE MAXIMUM SEEK TIME. THIS SEQUENCE IS REPEATED 128 TIMES (FOR
 *A TOTAL OF 256. SEEKS). MAXIMUM (FORWARD) TIME IS 55.0 MS.
 *
 *THERE IS NO SPECIFICATION GIVEN FOR THE MAXIMUM REVERSE
 *SEEK TIME ON THIS PARTICULAR DRIVE. THEREFORE, ANY REVERSE
 *SEEK TIMES ABOVE THE MAXIMUM TIME OF 55.0 MS WILL NOT BE
 *TYPED IN THE TIMING REPORT. HOWEVER, THE TIMING REPORT
 *WILL STILL TYPE THE MINIMUM, MAXIMUM AND AVERAGE TIMES
 *FOR THE REVERSE SEEKS.
 *

015704				15T15:	NOP		
015704	000240				BIT	BITS+<15*2>,15T15	;DO THIS TEST?
015706	033737	001572	001332		BNE	+6	;BR IF YES
015714	001002				JMP	15T16	;NO--JUMP TO TEST16
015716	000137	016454					
015722	012737	000015	001116		MOV	#15,\$15T15	;SET TEST #15 AND CLEAR (\$ERFLG)
015730	004737	030142			JSR	PC,LODPRM	;LOAD THE PARAMETERS FOR THE TEST
015734	012737	016114	001124		MOV	#TEST15,\$LPERR	;SETUP THE LOOP ON ERROR ADDRESS
015742	013737	002336	001220		MOV	RPT,\$TIMES	;GET THE ITERATION COUNT
015750	112737	000031	001131		MOV	#25,\$ERMAX	;MAX ERRORS ALLOWED FOR TEST
015756	012737	000015	001240		MOV	#15,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
319	015764	032777	010000	163162	BIT	#SW12,@SWR	;INHIBIT TYPING TEST NUMBER ?
	015772	001406			BEQ	+16	;BR IF YES

015774	104401	047617				TYPE	MSGTST	;TYPE 'TEST'
016000	013746	001240				MOV	\$TESTN,-(SP)	::SAVE \$TESTN FOR TYPEOUT
016004	104403					TYPOS		::GO TYPE--OCTAL ASCII
016006	003					.BYTE	3	::TYPE 3 DIGIT(S)
016007	000					.BYTE	0	::SUPPRESS LEADING ZEROS
320	016010	005737	001342			TST	CLKSTA	;KW11-P CLOCK
321	016014	003002				BGT	1\$	;YES--START TEST
324	016016	000137	016452			JMP	EXIT15	;NO--JUMP TO EXIT15
325	016022	012737	016022	001122	1\$:	MOV	#,\$LPADR	;SETUP THE LOOP ADDRESS
326	016030	004037	032552			JSR	R0,SRCH00	;DO A MASSBUS INIT & RECAL
327	016034	000402				BR	2\$	;RETURN HERE IF NO ERROR
339	016036	000137	016452			JMP	EXIT15	;RETURN HERE ON ERROR
	016042	012737	016042	001122	2\$:	MOV	#,\$LPADR	;ERROR LOOP ADDRESS
	016050	112737	000105	047020		MOVB	#SEEK,DPB.A+2	;SEEK=COMMAND
	016056	005037	047026			CLR	DPB.A+10	;USE TRACK 0 & SECTOR 0
	016062	013737	002340	047030		MOV	FC,DPB.A+12	;STARTING CYLINDER
	016070	012737	016452	001350		MOV	#EXIT15,BYPASS	;GO TO EXIT15 IF ERROR
	016076	004037	030402			JSR	R0,CALL.A	;GO EXECUTE THE COMMAND
	016102	012703	001700			MOV	#T12,R3	;PARAMETER POINTER
	016106	012737	016452	001222		MOV	#EXIT15,\$ESCAPE	::ESCAPE TO EXIT15 ON ERROR
	016114					TEST15:		
340	016114	012706	001100			MOV	#STACK,SP	;SETUP STACK
341	016120	012701	000200			MOV	#128.,R1	;REPEAT "'FC'-'LC'-'FC'" 128 TIMES
342	016124	004737	032736			JSR	PC,STRTMR	;INIT. THE TIMERS
343	016130	012777	016360	163352		MOV	#7\$,@PKV	;SETUP VECTOR IN CASE OF OVERFLOW
344	016136	012777	032734	022506		MOV	#DORT1,@RMVEC	;SETUP RM VECTOR
345	016144	005077	163346		1\$:	CLR	@PKB	;START COUNTING FROM ZERO
346	016150	013764	002342	000034		MOV	LC,RMDC(R4)	;MAXIMUM CYLINDER
347	016156	012764	000105	000000		MOV	#SEEK,RMCS1(R4)	;START A SEEK
348	016164	012777	000131	163322		MOV	#131,@PKCS	;START THE CLOCK
349	016172	000001				WAIT		;WAIT ON INTERRUPT
350	016174	042777	000101	163312		BIC	#101,@PKCS	;STOP THE CLOCK
351	016202	032764	040000	000012		BIT	#ERR,RMDS(R4)	;ERR=1?
352	016210	001411				BEQ	2\$	;NO--BRANCH
353	016212	104412				SAVREG		;SAVE R0-R5
	016214	012702	047076			MOV	#DTADPB,R2	;DPB POINTER
	016220	004737	045654			JSR	PC,SVRH70	;SAVE ALL THE RH/RM REGISTERS
	016224	004737	027334			JSR	PC,CNTCLR	;GO CLEAR MASSBUS CONTROLLER
	016230	104413				RESREG		;RESTORE R0-R5
354	016232	104017				EMT	17	;DISK ERROR OCCURRED
355	016234	005005			2\$:	CLR	R5	;SET THE UP/DOWN SWITCH TO UP
356	016236	004737	033000			JSR	PC,COUNT	;UP THE COUNT
357	016242	005077	163250			CLR	@PKB	;START COUNT AT ZERO
358	016246	013764	002340	000034		MOV	FC,RMDC(R4)	;BEGINNING CYLINDER
359	016254	012764	000105	000000		MOV	#SEEK,RMCS1(R4)	;START A SEEK
360	016262	012777	000131	163224		MOV	#131,@PKCS	;START THE CLOCK
361	016270	000001				WAIT		;WAIT ON INTERRUPT
362	016272	042777	000101	163214		BIC	#101,@PKCS	;STOP THE CLOCK
363	016300	032764	040000	000012		BIT	#ERR,RMDS(R4)	;ERR=1?
364	016306	001411				BEQ	3\$	;NO--BRANCH
365	016310	104412				SAVREG		;SAVE R0-R5
	016312	012702	047076			MOV	#DTADPB,R2	;DPB POINTER
	016316	004737	045654			JSR	PC,SVRH70	;SAVE ALL THE RH/RM REGISTERS
	016322	004737	027334			JSR	PC,CNTCLR	;GO CLEAR MASSBUS CONTROLLER
	016326	104413				RESREG		;RESTORE R0-R5
366	016330	104017				EMT	17	;DISK ERROR OCCURRED

```

367 016332 012705 177777 3$: MOV # -1,R5 ;SET THE UP/DOWN SWITCH TO DOWN
368 016336 004737 C33000 JSR PC,COUNT ;UPDATE THE COUNT
369 016342 005037 001416 CLR TIM.DN+2 ;FORGET ABOUT # OF SEEKS BELOW MINIMUM TIME
370 016346 005037 001422 CLR TIM.DN+6 ;FORGET ABOUT # OF SEEKS ABOVE MAXIMUM TIME
371 016352 005301 DEC R1 ;DONE?
372 016354 003273 BGT 1$ ;NO--BRANCH
373 016356 000420 BR 8$ ;YES--EXIT
374 016360 042777 000101 163126 7$: BIC #101,@PKCS ;STOP THE CLOCK
375 016366 005037 177776 CLR PS ;DROP THE PRIORITY
376 016372 012600 MOV (S)+,R0 ;PC OF WAIT+2
377 016374 005726 TST (SP)+ ;POP THE PS FROM THE STACK
378 016376 104412 SAVREG ;SAVE R0-R5
016400 012702 047076 MOV #DTADPB,R2 ;DPB POINTER
016404 004737 045654 JSR PC,SVRH70 ;SAVE ALL THE RH/RM REGISTERS
016410 004737 027334 JSR PC,CNTCLR ;GO CLEAR MASSBUS CONTROLLER
016414 104413 RESREG ;RESTORE R0-R5
379 016416 104020 EMT 20 ;CLOCK OVERFLOWED
380 016420 004737 027334 8$: JSR PC,CNTCLR ;GO CLEAR MASSBUS CONTROLLER
381 016424 004737 027426 JSR PC,ST.CLK ;INITIALIZE THE CLOCK
382 016430 012777 043306 022214 MOV #ISR,@RMVEC ;RESTORE RH/RM INT. VECTOR
383 016436 004037 033240 JSR R0,TYPTIM ;GO TYPE THE TIMES
016442 001700 T12 ;POINTER
384 016444 004037 033132 JSR R0,SPTYP ;
385 016450 001754 SP12 ;
386 016452 000004 EXIT15: SCOPE ;CALL SCOPE ROUTINE

```

1
2
3
4
5
6
7
8
19
20

.SBTTL ADDRESSING TESTS

[illegible]

```

:*****
: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 - 31 ARE WRITE CHECKED.
:THEN SECTOR 1 IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED.
:THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH
:REWRITE SECTOR 31 AND WRITE CHECK SECTORS 0-31.
:*****
TST16:

```

Line	Address	Op	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417
------	---------	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

39	016700	012706	001100		MOV	#STACK,SP	;LOAD THE STACK POINTER
40	016704	004037	031204		JSR	RO,DRVCAL	;START A DATA TRANSFER
	016710	012737	016710	001124	MOV	#, \$LPERR	;SETUP THE ERROR LOOP ADDRESS
	016716	012706	001100		MOV	#STACK,SP	;LOAD THE STACK POINTER
41	016722	004037	033766		JSR	RO,CLRBUF	;CLEAR BUFFER
42	016726	012737	000171	047100	MOV	#READ,DTADPB+2	;COMMAND = READ
43	016734	004037	031204		JSR	RO,DRVCAL	;START A DATA TRANSFER
44	016740	004037	034034		JSR	RO,CKSCTR	;CHECK THE SECTOR DATA READ
45	016744	012700	054522		MOV	#BUFFER,RO	;BUFFER ADDRESS
46	016750	005001			CLR	R1	;FIRST SECTOR
47	016752	012737	016752	001124	MOV	#, \$LPERR	;SETUP THE ERROR LOOP ADDRESS
	016760	012706	001100		MOV	#STACK,SP	;LOAD THE STACK POINTER
48	016764	012737	000161	047100	MOV	#WRITE,DTADPB+2	;COMMAND=WRITE DATA
49	016772	012737	177400	047102	MOV	#SCTRWC,DTADPB+4	;WORD COUNT
50	017000	010037	047104		MOV	RO,DTADPB+6	;BUFFER ADDRESS
51	017004	110137	047106		MOVB	R1,DTADPB+10	;SECTOR
52	017010	004037	031204		JSR	RO,DRVCAL	;START A DATA TRANSFER
53	017014	012737	017014	001124	MOV	#, \$LPERR	;SETUP THE ERROR LOOP ADDRESS
	017022	012706	001100		MOV	#STACK,SP	;LOAD THE STACK POINTER
54	017026	012737	000151	047100	MOV	#WRCKD,DTADPB+2	;COMMAND=WRITE CHECK DATA
55	017034	013737	001452	047102	MOV	TRCKWC,DTADPB+4	;WORD COUNT
56	017042	012737	054522	047104	MOV	#BUFFER,DTADPB+6	;BUFFER ADDRESS
57	017050	105037	047106		CLRB	DTADPB+10	;SECTOR
58	017054	004037	031204		JSR	RO,DRVCAL	;START A DATA TRANSFER
59	017060	062700	001000		ADD	#512.,RO	;MOVE TO NEXT SECTOR
60	017064	005201			INC	R1	
61	017066	023701	002470		CMP	PRMLMT+24,R1	;DONE?
62	017072	103334			BHIS	1\$	;NO--BRANCH
63	017074	000004			EXIT16:	SCOPE	;CALL SCOPE ROUTINE

 64
 76
 77

```

*****
*TEST 17 TRACK ADDRESSING TEST
*THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
*IN CYLINDER 'FC' SECTOR 'FS' OF EVERY TRACK WITH EACH TRACK
*GETTING ITS OWN TRACK ADDRESS.
*A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO ENSURE
*THE DATA IS VALID. THEN TRACK 0 IS REWRITTEN AND TRACK 1
*THROUGH LAST TRACK IS WRITE CHECKED. THEN TRACK 1 IS
*REWRITTEN AND TRACK 2 THROUGH LAST TRACK IS WRITE CHECKED.
*THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING NEXT TO LAST
*TRACK AND WRITE CHECKING LAST TRACK.
*****

```

TST17:

017076	000240			NOP			
017100	033737	001576	001332	BIT	BITS+<17*2>,TSTNMS	;DO THIS TEST?	
017106	001002			BNE	+.6	;BR IF YES	
017110	000137	017542		JMP	TST20	;NO--JUMP TO TEST20	
017114	012737	000017	001116	MOV	#17,\$TSTNM	;SET TEST #17 AND CLEAR (\$ERFLG)	
017122	004737	030142		JSR	PC,LODPKM	;LOAD THE PARAMETERS FOR THE TEST	
017126	012737	017216	001124	MOV	#TEST17,\$LPERR	;SETUP THE LOOP ON ERROR ADDRESS	
017134	013737	002336	001220	MOV	RPT,\$TIMES	;GET THE ITERATION COUNT	
017142	113737	001464	001131	MOVB	ERR,CT,\$ERMAX	;MAX ERRORS ALLOWED FOR TEST	
017150	012737	000017	001240	MOV	#17,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX	
78	017156	032777	010000	161770	BIT	#SW12,@SWR	;INHIBIT TYPING TEST NUMBER ?

017164	001406			BEG	.+16	:BR IF YES
017166	104401	047617		TYPE	MSGTST	:TYPE 'TEST'
017172	013746	001240		MOV	\$TESTN,-(SP)	:SAVE \$TESTN FOR TYPEOUT
017176	104403			TYPOS		:GO TYPE--OCTAL ASCII
017200	003			.BYTE	3	:TYPE 3 DIGIT(S)
017201	000			.BYTE	0	:SUPPRESS LEADING ZEROS
83	017202	012737	017540	001350	MOV	#EXIT17,BYPASS
	017210	012737	017216	001122	MOV	#TEST17,\$LPADR
	017216					:SETUP THE LOOP ADDRESS
84	017216	012706	001100		MOV	#STACK,SP
85	017222	004737	033730		JSR	PC,FILBUF
86	017226	012737	000161	047100	MOV	#WRITE,DTADPB+2
87	017234	013737	002340	047110	MOV	FC,DTADPB+12
88	017242	113737	002354	047106	MOVB	FS,DTADPB+10
89	017250	012737	177400	047102	MOV	#SCTRW,DTADPB+4
90	017256	012737	054522	047104	MOV	#BUFFER,DTADPB+6
91	017264	005000			CLR	R0
92	017266	012737	017266	001124	MOV	#,\$SLPERR
	017274	012706	001100		MOV	#STACK,SP
93						:TRACK=C
94	017300	110037	047107		MOVB	R0,DTADPB+11
95	017304	004037	031204		JSR	R0,DRVCAL
96	017310	062737	001000	047104	ADD	#256.*2.,DTADPB+6
97	017316	005200			INC	R0
98	017320	023700	001374		CMP	LSTRK,R0
99	017324	002365			BGE	1\$
100	017326	012737	054522	047104	MOV	#BUFFER,DTADPB+6
101	017334	005000			CLR	R0
102	017336	012737	017336	001124	MOV	#,\$SLPERR
	017344	012706	001100		MOV	#STACK,SP
103						:LOAD THE STACK POINTER
104	017350	012737	000151	047100	MOV	#WRCKD,DTADPB+2
105	017356	110037	047107		MOVB	R0,DTADPB+11
106	017362	004037	031204		JSR	R0,DRVCAL
107	017366	062737	001000	047104	ADD	#256.*2.,DTADPB+6
108	017374	005200			INC	R0
109	017376	023700	001374		CMP	LSTRK,R0
110	017402	002365			BGE	2\$
111	017404	005000			CLR	R0
112						:FIRST TRACK ADDRESS
113	017406	110037	047107		MOVB	R0,DTADPB+11
114	017412	010001			MOV	R0,R1
115	017414	012737	054522	047104	MOV	#BUFFER,DTADPB+6
116	017422	005301			DEC	R1
117	017424	002411			BLT	5\$
118	017426	062737	001000	047104	ADD	#256.*2.,DTADPB+6
119	017434	000772			BR	4\$
120	017436	012737	017436	001124	MOV	#,\$SLPERR
	017444	012706	001100		MOV	#STACK,SP
121	017450	012737	000161	047100	MOV	#WRITE,DTADPB+2
122	017456	004037	031204		JSR	R0,DRVCAL
123						:START A DATA TRANSFER
124	017462	062737	001000	047104	ADD	#256.*2.,DTADPB+6
125	017470	105237	047107		INCB	DTADPB+11
126	017474	012737	017474	001124	MOV	#,\$SLPERR
	017502	012706	001100		MOV	#STACK,SP
						:LOAD THE STACK POINTER

127	017506	012737	000151	047100	MOV	#WRCKD,DTADPB+2	;COMMAND=WRITE CHECK DATA
128	017514	004037	031204		JSR	R0,DRVCAL	;START A DATA TRANSFER
129	017520	123737	001374	047107	CMPB	LSTRK,DTADPB+11	;OUT OF TRACKS?
130	017526	003355			BGT	6\$	;NO--BRANCH
131	017530	005200			INC	R0	;NEXT TRACK TO WRITE
132	017532	023700	001374		CMP	LSTRK,R0	;OUT OF TRACKS?
133	017536	003323			BGT	3\$	;NO--BRANCH
134	017540	000004			EXIT17:	SCOPE	;CALL SCOPE ROUTINE
135							
155							
156							

```

*****
*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 177777 (ALL POSSIBLE PATTERNS)
*****

```

017542							
017542	000240				NOP		
017544	033737	001540	001334		BIT	BITS+<20*2-40>,TSTNMS+2	;DO THIS TEST ?
017552	001002				BNE	+6	;BR IF YES
017554	000137	020322			JMP	TST21	;NO--JUMP TO TEST21
017560	012737	000020	001116		MOV	#20,\$TSTNM	;SET TEST #20 AND CLEAR (\$ERFLG)
017566	004737	030142			JSR	PC,LODPRM	;LOAD THE PARMETERS FOR THE TEST
017572	012737	017770	001124		MOV	#TEST20,\$LPERR	;SETUP THE LOOP ON ERROR ADDRESS
017600	013737	002336	001220		MOV	RPT,\$TIMES	;GET THE ITERATION COUNT
017606	113737	001464	001131		MOVB	ERR,CT,\$ERMAX	;MAX ERRORS ALLOWED FOR TEST
017614	012737	000020	001240		MOV	#20,\$TESTN	;SET TEST NUMBER IN APT MAIL BOX
157	017622	032777	010000	161324	BIT	#SW12,@SWR	;INHIBIT TYPING TEST NUMBER ?
	017630	001406			BEO	+16	;BR IF YES
	017632	104401	047617		TYPE	,MSGTST	;TYPE 'TEST'
	017636	013746	001240		MOV	\$TESTN,-(SP)	;SAVE \$TESTN FOR TYPEOUT
	017642	104403			TYPOS		;GO TYPE--OCTAL ASCII
	017644	003			.BYTE	3	;TYPE 3 DIGIT(S)
	017645	000			.BYTE	0	;SUPPRESS LEADING ZEROS
158							
159							
160							
161	017646	012737	017646	001122	MOV	#,,\$LPADR	;SETUP THE LOOP ADDRESS
162	017654	005000			CLR	R0	;CLEAR SWITCH
163	017656	005004			CLR	R4	;FORM WORD COUNT IN R4

```

164 017660 013701 002356      MOV    LS,R1      ;LOAD LAST SECTOR
165 017664 163701 002354      SUB     FS,R1      ;COMPARE WITH FIRST SECTOR
166 017670 002004              BGE     1$          ;SKIP NEXT IF FS < OR = LS
167
168 017672 063701 002470      ADD     PRMLMT+24,R1 ;ADD MAXIMUM SECTOR ADDRESS TO
169 017676 005201              INC     R1          ;MAKE THE DIFFERENCE POSITIVE
170 017700 005100              COM     R0          ;SET SWITCH FOR FS > LS
171
172 017702 062704 000400      1$:    ADD     #256.,R4      ;WORDS/SECTOR
173 017706 005301              DEC     R1          ;LS - FS SECTORS MINUS ONE
174 017710 002374              BGE     1$          ;INCR. WORD COUNT IF NO. SECTORS STILL +
175 017712 005404              NEG     R4          ;FORM NEGATIVE WORD COUNT
176 017714 010405              MOV     R4,R5        ;COPY NORMAL WORD COUNT INTO SMALL WC
177 017716 005700              TST     R0          ;FS > LS SWITCH SET?
178 017720 001412              BEQ     3$          ;NO, SKIP NEXT
179
180 017722 005005              CLR     R5          ;FORM WORD COUNT FOR FS > LS
181 017724 013701 002470      MOV     PRMLMT+24,R1 ;MAX SECTOR ADDRESS
182 017730 163701 002354      SUB     FS,R1      ;SUBTRACT THE FIRST SECTOR
183 017734 062705 000400      2$:    ADD     #256.,R5        ;WORDS/SECTOR
184 017740 005301              DEC     R1          ;NO SECTORS MINUS ONE
185 017742 002374              BGE     2$          ;INCR. WORD COUNT IF NO. SECTORS STILL +
186 017744 005405              NEG     R5          ;FORM NEGATIVE WORD COUNT FOR THIS CASE
187
188                          ;SET UP FOR DATA TRANSFERS AND PATTERN VARIATIONS
189
190 017746 113737 002354 047106 3$:    MOV     FS,DTADPB+10 ;SECTOR
191 017754 012737 054522 047104      MOV     #BUFFER,DTADPB+6 ;DATA BUFFER
220 017762 012737 020320 001350      MOV     #EXIT20,BYPASS
221 017770
222 017770 012706 001100      TEST20: MOV    #STACK,SP      ;LOAD THE STACK POINTER
017774 005037 001434      CLR     WCEFLG ;CLEAR THE WRITE CHECK ERROR FLAG
020000 013701 002340      MOV     FC,R1      ;PICKUP FIRST CYLINDER
020004 000407              BR      2$          ;SKIP PATTERN SET-UP FIRST TIME THRU

020006 005720              1$:    TST     (R0)+      ;MOVE TO NEXT DATA PATTERN
020010 022700 000040      CMP     #16.*2.,R0 ;OUT OF PATTERNS?
020014 003004              BGT     3$          ;NO, STAY ON THIS CYLINDER UNTIL DONE
020016 004037 033654      JSR     R0,INCCYL ;MOVE TO NEXT CYLINDER
020022 000536              BR      EXIT20      ;OUT OF CYLINDERS

                          ;DO NEXT CYLINDER

020024 005000              2$:    CLR     R0          ;START WITH PATTERN 0
020026 036037 001540 002360 3$:    BIT     BITS(R0),PAT ;THIS PATTERN SELECTED?
020034 001764              BEQ     1$          ;NO, GO BACK AND GET ONE THAT WAS
020036 013702 002346      MOV     FT,R2          ;FIRST TRACK
020042 010137 047110      MOV     R1,DTADPB+12 ;LOAD THE CYLINDER
020046 110237 047107      4$:    MOV     R2,DTADPB+11 ;LOAD THE TRACK

020052 020127 001466      CMP     R1,#822.      ;CHECK FOR LAST DISK CYLINDER (DEC144 FILE)
020056 001003              BNE     10$         ;SKIP LAST TRACK CHECK IF NOT
020060 020237 001574      CMP     R2,LSTRK    ;LAST DISK TRACK ?
020064 001515              BEQ     EXIT20      ;DON'T TEST LAST TRACK AS IT HAS BAD
                          ;BLOCK INFORMATION STORED ON IT

221 020066 010437 047102      10$:   MOV     R4,DTADPB+4 ;LOAD THE WORD COUNT
222 020072 023701 002342      CMP     LC,R1      ;LAST DISK CYLINDER TO TEST ?

```

```

223 020076 003005          BGT      5$          ;NO, SKIP TRACK CHECK
224 020100 023702 001374    CMP      LSTRK,R2      ;LAST DISK TRACK TO TEST?
225 020104 002002          BGE      5$          ;NO, SKIP NEXT
226 020106 010537 047102    MOV      R5,DTADPB+4    ;SHORT WORD COUNT
227 020112 017703 161036    5$:      MOV      @SWR,R3      ;INHIBIT WRITE AND
228 020116 005103          COM      R3              ;WRITE CHECK?
229 020120 032703 000030    BIT      #SW04!SW03,R3
230 020124 001436          BEQ      7$          ;YES--BRANCH
231 020126 004737 034404    JSR      PC,SETBUF      ;MOVE DATA PATTERN INTO THE BUFFER
232 020132 032777 000020 161014    BIT      #SW04,@SWR      ;INHIBIT WRITE?
233 020140 001012          BNE      6$          ;YES, DO NEXT PORTION OF TESTING
234 020142 012737 020142 001124    MOV      #.,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
235 020150 012706 001100    MOV      #STACK,SP      ;LOAD THE STACK POINTER
235 020154 012737 000161 047100    MOV      #WRITE,DTADPB+2 ;COMMAND=WRITE DATA
236
237
238
239 020162 004037 031204          JSR      R0,DRVCL      ;START A DATA TRANSFER
240 020166 032777 000010 160760 6$:    BIT      #SW03,@SWR      ;INHIBIT WRITE CHECK?
241 020174 001012          BNE      7$          ;YES--BRANCH
242 020176 012737 020176 001124    MOV      #.,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
243 020204 012706 001100    MOV      #STACK,SP      ;LOAD THE STACK POINTER
243 020210 012737 000151 047100    MOV      #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
244 020216 004037 031204          JSR      R0,DRVCL      ;START A DATA TRANSFER
245 020222 005737 001434          7$:    TST      WCEFLG ;WRITE CHECK ERROR FLAG SET?
246 020226 001404          BEQ      8$          ;NO--BRANCH
247 020230 032777 000001 160716    BIT      #SW00,@SWR      ;PERFORM READ AFTER 'ATAL 'WCE'?
248 020236 001424          BEQ      9$          ;NO--BRANCH
249 020240 032777 000004 160706 8$:    BIT      #SW02,@SWR      ;INHIBIT READ DATA AND SOFTWARE COMPARE?
250 020246 001020          BNE      9$          ;YES--BRANCH
251 020250 012737 020250 001124    MOV      #.,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
252 020256 012706 001100    MOV      #STACK,SP      ;LOAD THE STACK POINTER
253 020262 012737 000171 047100    MOV      #READ,DTADPB+2 ;COMMAND=READ
254 020270 004037 031204          JSR      R0,DRVCL      ;START A DATA TRANSFER
255 020274 032777 000002 160652    BIT      #SW01,@SWR      ;COMPARE THE DATA?
256 020302 001002          BNE      9$          ;NO--BRANCH
257 020304 004737 034474          JSR      PC,DATCMP      ;YES--DO IT
258 020310 004037 033624          9$:    JSR      R0,INCTRK      ;MOVE TO NEXT TRACK
259 020314 000634          BR      1$          ;OUT OF TRACKS GO TO NEXT PATTERN
260 020316 000653          BR      4$          ;LOOP
260 020320 000004          EXIT20: SCOPE          ;CALL SCOPE ROUTINE
261
262
263
264
265
266
267

```

```

*****
*TEST 21      RANDOM ADDRESS AND RANDOM PATTERN TEST
*STARTING AT 'FC' AND GOING SEQUENTIALLY TO '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 1000.
*
*1)  GENERATE A RANDOM SECTOR ADDRESS
*2)  WRITE A RANDOM PATTERN AT THE ADDRESS
*    GENERATED IN 1.
*3)  GENERATE A NEW RANDOM SECTOR ADDRESS

```

```

;*4) READ THE SECTOR AT THE ADDRESS
;*   GENERATED IN 3.
;*5) DO A SOFTWARE CHECK OF THE DATA READ IN 4 AGAINST
;*   THE ORIGINAL RANDOM PACK DATA.
;*6) DO A WRITE CHECK OF THE DATA WRITTEN IN 2
;*7) GENERATE A NEW RANDOM SECTOR 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:
020322
020322 000240
020324 033737 001542 001334
020332 001002
020334 000137 021140
                                NOP
                                BIT      BITS+<21*2-40>,TSTNMS+2 ;DO THIS TEST ?
                                BNE      .+6 ;BR IF YES
                                JMP      TST22 ;NO--JUMP TO TEST22

020340 012737 000021 001116
020346 004737 030142
020352 012737 020616 001124
020360 013737 002336 001220
020366 113737 001464 001131
020374 012737 000021 001240
                                MOV      #21,$TSTNM ;SET TEST #21 AND CLEAR ($ERFLG)
                                JSR      PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
                                MOV      #TST21,$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
                                MOV      RPT,$TIMES ;GET THE ITERATION COUNT
                                MOVB     ERR,CT,$ERMAX ;MAX ERRORS ALLOWED FOR TEST
                                MOV      #21,$TESTN ;SET TEST NUMBER IN APT MAIL BOX

288 020402 032777 010000 160544
020410 001406
020412 104401 047617
020416 013746 001240
020422 104403
020424 003
020425 000
                                BIT      #SW12,$SWR ;INHIBIT TYPING TEST NUMBER ?
                                BEQ      .+16 ;BR IF YES
                                TYPE     ,MSGTST ;TYPE 'TEST'
                                MOV      $TESTN,-(SP) ;SAVE $TESTN FOR TYPEOUT
                                TYPOS    ;GO TYPE--OCTAL ASCII
                                .BYTE    3 ;TYPE 3 DIGIT(S)
                                .BYTE    0 ;SUPPRESS LEADING ZEROS

289 020426 012737 020426 001122
292 020434 012737 021136 001350
293 020442 012737 176543 026524
294 020450 012737 123456 026526
295 020456 013737 002340 047110
296 020464 013737 001452 047102
297 020472 012737 054522 047104
298 020500 012737 000161 047100
299 020506 032737 100000 001314
300 020514 001027
301 020516 004037 035012
302 020522 005037 047106
303 020526 012737 020526 001124
020534 012706 001100
                                MOV      #.,$LPADR ;SETUP THE LOOP ADDRESS
                                MOV      #EXIT21,BYPASS
                                MOV      #176543,$SHNUM ;PRIME THE RANDOM NUMBER GENERATOR
                                MOV      #123456,$LONUM
                                MOV      FC,DTADPB+12 ;CYLINDER
                                MOV      TRCKWC,DTADPB+4 ;WORD COUNT FOR 32/30 SECTORS (FULL TRACK)
                                MOV      #BUFFER,DTADPB+6 ;BUFFER ADDRESS
                                MOV      #WRITE,DTADPB+2 ;COMMAND
                                BIT      #SW15,C.SWR ;WRITE THE DISK PACK BEFORE TESTING?
                                BNE      3$ ;NO--BEGIN TESTING
                                JSR      RO,FILRAN ;FILL DATA BUFFER WITH RANDOM DATA
                                CLR      DTADPB+10 ;SECTOR AND TRACK
                                MOV      #.,$LPERR ;SETUP THE ERROR LOOP ADDRESS
                                MOV      #STACK,SP ;LOAD THE STACK POINTER

304 020540
020540 004037 031204
305 020544 105237 047107
306 020550 123737 001374 047107
307 020556 002370
308 020560 005237 047110
309 020564 023737 002342 047110
310 020572 002353
311
312
313 020574 012737 177400 047102 3$:
                                MOV      #SCTRWC,DTADPB+4 ;WORD COUNT

```

```

318 020602 012737 020616 001122      MOV      #TEST21,$LPADR
      020610 012737 020616 001124      MOV      #TEST21,$LPERR
      020616                                TEST21:
319 020616 012706 001100      MOV      #STACK,SP          ;SET STACK POINIER
320 020622 004037 035266      JSR      RO,RANADR          ;GENERATE A RANDOM ADDRESS
321 020626 013737 047106 001440      MOV      DTADPB+0,SVADR      ;SAVE THE TRACK/SECTOR
322 020634 013737 047110 001442      MOV      DTADPB+12,SVADR+2      ;SAVE THE CYLINDER
323 020642 012737 000161 047100      MOV      #WRITE,DTADPB+2      ;COMMAND=WRITE DATA
324 020650 012701 054522      MOV      #BUFFER,R1          ;WRITE BUFFER ADDRESS FOR 'RANPAT'
325 020654 010137 047104      MOV      R1,DTADPB+6      ;INTO THE DATA PARAMETER BLOCK
326 020660 004037 035232      JSR      RO,RANPAT          ;GENERATE RANDOM 256 WORD PATTERN
327                                ;AND PUT INTO THE WRITE BUFFER
328 020664 012737 020664 001124      MOV      #,$LPERR          ;SETUP THE ERROR LOOP ADDRESS
      020672 012706 001100      MOV      #STACK,SP          ;LOAD THE STACK POINTER
329 020676 004037 031204      JSR      RO,DRVCL          ;START A DATA TRANSFER
330
331 020702 004037 035266      JSR      RO,RANADR          ;GENERATE A NEW RANDOM ADDRESS
332 020706 012737 000171 047100      MOV      #READ,DTADPB+2      ;COMMAND=READ DATA
333 020714 012737 055522 047104      MOV      #BUFFER+512.,DTADPB+6      ;READ BUFFER ADDRESS
334 020722 012737 020722 001124      MOV      #,$LPERR          ;SETUP THE ERROR LOOP ADDRESS
      020730 012706 001100      MOV      #STACK,SP          ;LOAD THE STACK POINTER
335 020734 004037 031204      JSR      RO,DRVCL          ;START A DATA TRANSFER
336 020740 005737 001466      TST      BASFLG          ;IF BAD SECTOR ENCOUNTERED,SKIP NEXT CALL
337 020744 100402              BMI      .+6              ;DON'T COMPARE DATA
338 020746 004037 035034      JSR      RO,RANCK          ;SOFTWARE CHECK THE DATA
339
340 020752 013737 001440 047106      MOV      SVADR,DTADPB+10      ;GET ADDRESS OF WHERE THE LAST
341 020760 013737 001442 047110      MOV      SVADR+2,DTADPB+12      ;WRITE WAS PERFORMED
342 020766 012737 000151 047100      MOV      #WRCKD,DTADPB+2      ;COMMAND=WRITE CHECK DATA
343 020774 012737 054522 047104      MOV      #BUFFER,DTADPB+6      ;DATA BUFFER ADDRESS FOR HARDWARE
344                                ;CHECK OF THE DATA
345 021002 012737 021002 001124      MOV      #,$LPERR          ;SETUP THE ERROR LOOP ADDRESS
      021010 012706 001100      MOV      #STACK,SP          ;LOAD THE STACK POINTER
346 021014 004037 031204      JSR      RO,DRVCL          ;START A DATA TRANSFER
347
348 021020 004037 035266      JSR      RO,RANADR          ;GENERATE A NEW RANDOM ADDRESS
349 021024 012737 000171 047100      MOV      #READ,DTADPB+2      ;COMMAND=READ
350 021032 012737 055522 047104      MOV      #BUFFER+512.,DTADPB+6      ;DATA BUFFER ADDRESS
351 021040 012737 021040 001124      MOV      #,$LPERR          ;SETUP THE ERROR LOOP ADDRESS
      021046 012706 001100      MOV      #STACK,SP          ;LOAD THE STACK POINTER
352 021052 004037 031204      JSR      RO,DRVCL          ;START A DATA TRANSFER
353 021056 005737 001466      TST      BASFLG          ;ENCOUNTER BAD SECTOR ?
354 021062 100402              BMI      .+6              ;DON'T COMPARE DATA IF SO
355 021064 004037 035034      JSR      RO,RANCK          ;SOFTWARE CHECK THE DATA
356
357 021070 013737 001440 047106      MOV      SVADR,DTADPB+10      ;GET DISK ADDRESS OF THE
358 021076 013737 001442 047110      MOV      SVADR+2,DTADPB+12      ;LAST WRITE
359 021104 012737 000151 047100      MOV      #WRCKD,DTADPB+2      ;COMMAND=WRITE CHECK DATA
360 021112 012737 054522 047104      MOV      #BUFFER,DTADPB+6      ;DATA BUFFER ADDRESS
361 021120 012737 021120 001124      MOV      #,$LPERR          ;SETUP THE ERROR LOOP ADDRESS
      021126 012706 001100      MOV      #STACK,SP          ;LOAD THE STACK POINTER
362 021132 004037 031204      JSR      RO,DRVCL          ;START A DATA TRANSFER
363 021136 000004      EXIT21: SCOPE          ;CALL SCOPE ROUTINE
364
371
372

```

 :*TEST 22 SEEK TIME ADJUSTMENT TEST

;*THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 255 TO ALLOW THE
 ;*OPERATOR TO ADJUST THE AVERAGE SEEK TIME ON AN RM05/3/2 USING THE
 ;*DDU. THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
 ;*SO THAT THE AVERAGE SEEK TIME INDICATORS ON THE DDU MAY BE OBSERVED.
 ;*****

021140				TST22:	NOP	
021140	000240				BIT	BITS+<22*2-40>,TSTNMS+2 ;DO THIS TEST ?
021142	033737	001544	001334		BNE	.+6 ;BR IF YES
021150	001002				JMP	\$EJP ;NO--GO TO THE END OF THE PROGRAM
021152	000137	021334				
021156	012737	000022	001116		MOV	#22,\$TSTNM ;SET TEST #22 AND CLEAR (\$ERFLG)
021164	004737	030142			JSR	PC,LODPRM ;LOAD THE PARMETERS FOR THE TEST
021170	012737	021252	001124		MOV	#TEST22,\$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
021176	013737	002336	001220		MOV	RPT,\$TIMES ;GET THE ITERATION COUNT
021204	112737	000144	001131		MOVB	#100,\$ERMAX ;MAX ERRORS ALLOWED FOR TEST
021212	012737	000022	001240		MOV	#22,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
373						
021220	032777	010000	157726		BIT	#SW12,@SWR ;INHIBIT TYPING TEST NUMBER ?
021226	001406				BEQ	.+16 ;BR IF YES
021230	104401	047617			TYPE	,MSGTST ;TYPE 'TEST'
021234	013746	001240			MOV	\$TESTN,-(SP) ;SAVE \$TESTN FOR TYPEOUT
021240	104403				TYPOS	;;GO TYPE--OCTAL ASCII
021242	003				.BYTE	3 ;TYPE 3 DIGIT(S)
021243	000				.BYTE	0 ;SUPPRESS LEADING ZEROS
377	021244	012737	021252	001122	MOV	#TEST22,\$LPADR ;SETUP THE LOOP ADDRESS
	021252					
378	021252	012706	001100		MOV	#STACK,SP ;SETUP THE STACK POINTER
379	021256	013737	002342	047030	MOV	LC,DPB.A+12 ;ENDING CYLINDER
380	021264	112737	000105	047020	MOVB	#SEEK,DPB.A+2 ;SEEK=COMMAND
381	021272	004037	030402		JSR	RO,CALL.A ;GO EXECUTE THE COMMAND
382	021276	004037	032326		JSR	RO,STALL ;STALL
383	021302	001460			.WORD	STALL3 ;ADDRESS OF STALL VALUE
384	021304	013737	002340	047030	MOV	FC,DPB.A+12 ;STARTING CYLINDER
385	021312	112737	000105	047020	MOVB	#SEEK,DPB.A+2 ;SEEK=COMMAND
386	021320	004037	030402		JSR	RO,CALL.A ;GO EXECUTE THE COMMAND
387	021324	004037	032326		JSR	RO,STALL ;STALL
388	021330	001460			.WORD	STALL3 ;ADDRESS OF STALL VALUE
389	021332	000004			EXIT22:	SCOPE ;CALL SCOPE ROUTINE
390						
395						

.SBTTL END OF PASS ROUTINE

 *INCREMENT THE PASS NUMBER (\$PASS)
 *INDICATE END-OF-PROGRAM AFTER 8. PASSES THRU THE PROGRAM
 *IF THERES A MONITOR GO TO IT
 *IF THERE ISN'T JUMP TO RETURN

021334			\$EOP:	TYPE	,65\$	::TYPE ASCII STRING
021334	104401	021342		BR	64\$	::GET OVER THE ASCII
021340	000407		::65\$:	.ASCII	<CRLF><LF>/END OF PASS/	
021360			64\$:	TYPE	,67\$	::TYPE ASCII STRING
021360	104401	021366		BR	66\$	::GET OVER THE ASCII
021364	000405		::67\$:	.ASCII	/ ON DRIVE/	
021400			66\$:	MOV	CHKDRV,-(SP)	::SAVE CHKDRV FOR TYPEOUT
021400	013746	001352		TYPOS		::GO TYPE--OCTAL ASCII
021404	104403			.BYTE	2	::TYPE 2 DIGIT(S)
021406	002			.BYTE	0	::SUPPRESS LEADING ZEROS
021407	000			TST	\$ERTTL	::ANY ERRORS DETECTED ?
021410	005737	001126		BEQ	1\$	::BR IF NO
021414	001420			TYPE	,69\$	::TYPE ASCII STRING
021416	104401	021424		BR	68\$	::GET OVER THE ASCII
021422	000412		::69\$:	.ASCII	/ ERRORS DETECTED=	
021450			68\$:	MOV	\$ERTTL,-(SP)	::SAVE \$ERTTL FOR TYPEOUT
021450	013746	001126		TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
021454	104402		1\$:	CLR	\$ERTTL	::ZERO ERROR TOTAL
021456	005037	001126		CLR	\$TSTNM	::ZERO THE TEST NUMBER
021462	005037	001116		CLR	\$TIMES	::ZERO THE NUMBER OF ITERATIONS
021466	005037	001220		INC	\$PASS	::INCREMENT THE PASS NUMBER
021472	005237	001242		BIC	#100000,\$PASS	::DON'T ALLOW A NEG. NUMBER
021476	042737	100000 001242		DEC	(PC)+	::LOOP?
021504	005327		\$EOPCT:	.WORD	8.	
021506	000010			BGT	\$DOAGN	::YES
021510	003027			MOV	(PC)+,@(PC)+	::RESTORE COUNTER
021512	012737		\$ENDCT:	.WORD	8.	
021514	000010			\$EOPCT		
021516	021506			TYPE	,65\$	::TYPE ASCII STRING
021520	104401	021526		BR	64\$	::GET OVER THE ASCII
021524	000407		::65\$:	.ASCII	<CRLF>/END OF TEST/<CRLF>	
021544			64\$:	TYPE	,\$ENULL	::TYPE NULL CHARACTER
021544	104401	021574	\$GET42:	MOV	@42,R0	::GET MONITOR ADDRESS
021550	013700	000042		BEQ	\$DOAGN	::BRANCH IF NO MONITOR
021554	001405			RESET		::CLEAR THE WORLD
021556	000005		\$ENDAD:	JSR	PC,(R0)	::GO TO MONITOR
021560	004710			NOP		::SAVE ROOM
021562	000240			NOP		::FOR
021564	000240			NOP		::ACT11
021566	000240		\$DOAGN:			
021570				JMP	@(PC)+	::RETURN
021570	000137		\$RTNAD:	.WORD	RTURN	
021572	021600		\$ENULL:	.BYTE	-1,-1,0	::NULL CHARACTER STRING
021574	377	377 000		.EVEN		

2					
3	021600	012706	001100	RTURN: MOV	#STACK,SP ;RESTORE STACK
4	021604	004737	024060	JSR	PC,STKINT ;MAKE SURE KEYBOARD INTERRUPT AND
5	021610	004737	027426	JSR	PC,ST.CLK ;INITIALIZE THE CLOCK
6	021614	000137	006662	JMP	RSTART ;RETURN TO RESTART

.SBTTL ERROR HANDLER ROUTINE

 *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 *AND GO TO TYPERR ON ERROR
 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 *SW15=1 HALT ON ERROR
 *SW13=1 INHIBIT ERROR TYPEOUTS
 *SW10=1 BELL ON ERROR
 *SW09=1 LOOP ON ERROR
 *CALL
 * ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

021620	105037	022260		\$ERROR:	CLRB	IBSAVE	;;CLEAR THE ITEM BYTE SAVE LOCATION
021624	104407				CKSWR		;;TEST FOR CHANGE IN SOFT-SWR
021626	032777	000400	157320		BIT	#SW08,@SWR	;;SEND ERROR MESSAGE TO TTY?
021634	001411				BEQ	7\$	;;YES--BRANCH
021636	005737	001326			TST	LPTAVL	;;IS THERE A LINE PRINTER AVAILABLE?
021642	001406				BEQ	7\$	;;NO--BRANCH
021644	013737	001534	001164		MOV	LPS,\$TPS	;;YES--SETUP STATUS
021652	013737	001536	001166		MOV	LPB,\$TPB	;;AND BUFFER REG.'S FOR LINE PRINTER
021660	105237	001117		7\$:	INCB	\$ERFLG	;;SET THE ERROR FLAG
021664	001775				BEQ	7\$	;;DON'T LET THE FLAG GO TO ZERO
021666	013777	001116	157262		MOV	\$TSTNM,@DISPLAY	;;DISPLAY TEST NUMBER AND ERROR FLAG
021674	032777	002000	157252		BIT	#BIT10,@SWR	;;BELL ON ERROR?
021702	001402				BEQ	1\$	;;NO - SKIP
021704	104401	001224			TYPE	\$BELL	;;RING BELL
021710	005237	001126		1\$:	INC	\$ERTTL	;;COUNT THE NUMBER OF ERRORS
021714	011637	001132			MOV	(SP),\$ERRPC	;;GET ADDRESS OF ERROR INSTRUCTION
021720	162737	000002	001132		SUB	#2,\$ERRPC	
021726	117737	157200	001130		MOVB	@\$ERRPC,\$ITEMB	;;STRIP AND SAVE THE ERROR ITEM CODE
021734	032777	001000	157212		BIT	#BIT09,@SWR	;;SEE IF LOOP ON ERROR IS SET
021742	001060				BNE	1004\$	;;BRANCH AROUND ROUTINE IF SO
021744	122737	000177	001130		CMPSB	#177,\$ITEMB	;;SEE IF THIS IS THE POWER FAIL CALL
021752	001454				BEQ	1004\$	;;BRANCH AROUND ROUTINE IF IT IS
021754	105737	022260			TSTB	IBSAVE	;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
021760	001047				BNE	1003\$	;;BRANCH IF SO
021762	022737	177777	022256		CMP	#-1,CPSAVE	;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
021770	001445				BEQ	1004\$	;;BRANCH IF SO
021772	013746	000004			MOV	ERRVEC,-(SP)	;;SAVE CONTENTS OF ERROR VECTOR
021776	012737	022014	000004		MOV	#1000\$,ERRVEC	;;SETUP 'TRAP' RETURN ADDRESS
022004	013737	177766	022256		MOV	177766,CPSAVE	;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
022012	000406				BR	1001\$	
022014	012737	177777	022256	1000\$:	MOV	#-1,CPSAVE	;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
022022	012716	022030			MOV	#1001\$,(SP)	;;SETUP RETURN ADDRESS
022026	000002				RTI		
022030	012637	000004		1001\$:	MOV	(SP)+,ERRVEC	;;RESTORE CONTENTS OF ERROR VECTOR
022034	022737	177777	022256	1002\$:	CMP	#-1,CPSAVE	;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
022042	001420				BEQ	1004\$	;;BRANCH IF SO
022044	032737	000001	022256		BIT	#BIT00,CPSAVE	;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
022052	001414				BEQ	1004\$	;;BRANCH IF OK
022054	042737	000001	177766		BIC	#BIT00,177766	;;CLEAR THE BIT FOUND SET
022062	113737	001130	022260		MOVB	\$ITEMB,IBSAVE	;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
022070	112737	000177	001130		MOVB	#177,\$ITEMB	;;SET \$ITEMB TO SPECIAL POWER FAIL POINTER
022076	000402				BR	1004\$	;;BRANCH OVER IBSAVE CLEARING

022100	105037	022260	1003\$:	CLRB	IBSAVE	::CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
022104			1004\$:			
022104	032777	020000	157042	BIT	#BIT13,@SWR	::SKIP TYPEOUT IF SET
022112	001004			BNE	20\$	::SKIP TYPEOUTS
022114	004737	022262		JSR	PC,TYPERR	::GO TO USER ERROR ROUTINE
022120	104401	001231		TYPE	,\$CRLF	
022124			20\$:			
022124	122737	000001	001254	CMPB	#APTENV,\$ENV	::RUNNING IN APT MODE
022132	001007			BNE	2\$	::NO,SKIP APT ERROR REPORT
022134	113737	001130	022146	MOVB	\$ITEMB,21\$	::SET ITEM NUMBER AS ERROR NUMBER
022142	004737	026766		JSR	PC,\$ATY4	::REPORT FATAL ERROR TO APT
022146	000		21\$:	.BYTE	0	
022147	000			.BYTE	0	
022150	000777		22\$:	BR	22\$	::APT ERROR LOOP
022152	105737	022260	2\$:	TSTB	IBSAVE	::SEE IF IBSAVE IS LOADED
022156	001005			BNE	3\$	::BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
022160	005777	156770		TST	@SWR	::HALT ON ERROR
022164	100002			BPL	3\$	::SKIP IF CONTINUE
022166	000000			HALT		::HALT ON ERROR!
022170	104407			CKSWR		::TEST FOR CHANGE IN SOFT-SWR
022172			3\$:			
022172	105737	022260		TSTB	IBSAVE	::SEE IF ITEM BYTE SAVE LOCATION HAS AN ERROR CALL
022176	001230			BNE	7\$	::BRANCH BACK TO CALL ORIGINAL ERROR
022200	032777	001000	156746	BIT	#BIT09,@SWR	::LOOP ON ERROR SWITCH SET?
022206	001402			BEQ	4\$	::BR IF NO
022210	013716	001124		MOV	\$LPERR,(SP)	::FUDGE RETURN FOR LOOPING
022214	005737	001222	4\$:	TST	\$ESCAPE	::CHECK FOR AN ESCAPE ADDRESS
022220	001402			BEQ	5\$	::BR IF NONE
022222	013716	001222		MOV	\$ESCAPE,(SP)	::FUDGE RETURN ADDRESS FOR ESCAPE
022226			5\$:			
022226	022737	021560	000042	CMP	#SENDAD,@#42	::ACT-11 AUTO-ACCEPT?
022234	001001			BNE	6\$	::BRANCH IF NO
022236	000000			HALT		::YES
022240			6\$:			
022240	013737	001530	001164	MOV	TPS,\$TPS	::SET STATUS AND BUFFER REG.'S
022246	013737	001532	001166	MOV	TPB,\$TPB	::FOR TTY
022254	000002			RTI		::RETURN FROM ERROR CALL
022256	000000			CPSAVE: .WORD	0	::LOCATION TO SAVE CPU ERROR REG CONTENTS
022260	000000			IBSAVE: .WORD	0	::LOCATION TO SAVE ITEM BYTE

.SBTTL TYPERR - TYPE ERROR ROUTINE

```
2
3
4
5
6
7
8
9
10
11 022262 113737 001116 001212 TYPERR: MOVB $STNM,$TMP0 ;SAVE THE TEST NUMBER
12 022270 104412 SAVREG ;SAVE R0 - R5
13 022272 162700 000004 SUB #4,R0 ;FORM TEST PC
14 022276 010037 001176 MOV R0,$REG0 ;COPY R0-R5 IN $REG0-$REG5
15 022302 010137 001200 MOV R1,$REG1
16 022306 010237 001202 MOV R2,$REG2
17 022312 010337 001204 MOV R3,$REG3
18 022316 010437 001206 MOV R4,$REG4
19 022322 010537 001210 MOV R5,$REG5
20 022326 113700 001130 MOVB $ITEMB,R0 ;PICKUP ERROR ITEM NUMBER
21 022332 122700 000177 CMPB #177,R0 ;SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
22 022336 001007 BNE 1$ ;BRANCH IF NOT
23 022340 005001 CLR R1
27 022342 013737 001240 023040 MOV $TESTN,PFTSTN ;GET TEST NUMBER
29 022350 012700 022700 MOV #PFECB,R0 ;MOVE POWER FAIL ERROR CALL TABLE TO R0
30 022354 000412 BR 3$
31 022356 010001 1$: MOV R0,R1 ;AND COPY IT INTO R1
32 022360 005300 DEC R0 ;FORM INDEX FOR ERROR TABLE
33 022362 106300 ASLB R0
34 022364 106300 ASLB R0
35 022366 106300 ASLB R0
36 022370 103002 BCC 2$ ;IS ERROR > 37?
37 022372 062700 000240 ADD #ITEM41-$ERRTB,R0 ;YES--FORM OFFSET
38 022376 062700 002014 2$: ADD #ERRTB,R0 ;FORM ADDRESS
39 022402 012037 022422 3$: MOV (R0)+,4$ ;GET ERROR MESSAGE (EM) POINTER
40 022406 001451 BEQ 9$ ;BRANCH IF THERE ISN'T ONE
41 022410 004737 027216 JSR PC,INCEC ;INCREMENT ERROR COUNT
42 022414 104401 001231 TYPE $CRLF ;'CARRIAGE RETURN - LINE FEED'
43 022420 104401 TYPE
44 022422 000000 4$: .WORD 0 ;'EM' POINTER GOES HERE
45 022424 162701 000041 SUB #41,R1 ;SPECIAL ERROR ITEM NUMBER?
46 022430 100440 BMJ 9$ ;NO--BRANCH
47 022432 013701 001356 MOV SVSTAT,R1 ;GET STATUS/ERROR INDICATOR
48 022436 106301 ASLB R1 ;STRIP 'DONE' BIT (BIT07)
49 022440 006301 ASL R1 ;STRIP 'ERROR' BIT (BIT15)
50 022442 012702 001762 MOV #STATBL,R2 ;1ST ADDRESS ON STATUS MESSAGE POINTERS
51 022446 005003 CLR R3 ;CARRIAGE RETURN-LINE FEED SWITCH
52 022450 104401 022456 TYPE ,65$ ;TYPE ASCIZ STRING
53 022454 000402 BR 64$ ;GET OVER THE ASCIZ
54 022462 012237 022504 64$: .ASCIZ / (/
55 022466 006301 5$: MOV (R2)+,7$ ;MESSAGE POINTER
56 022470 103013 ASL R1 ;TYPE THIS MESSAGE?
57 022472 005103 BCC 8$ ;NO--BRANCH
58 022474 001002 COM R3 ;YES--TYPE A 'CR' & 'LF'?
59 022476 104401 001231 BNE 6$ ;NO--BRANCH
60 TYPE $CRLF ;YES
```

59	022502	104401		6\$:	TYPE			
60	022504	000000		7\$:	.WORD	0		;MESSAGE POINTER GOES HERE
61	022506	005701			TST	R1		;MORE TO TYPE?
62	022510	001403			BEQ	8\$		;NO--BRANCH
63	022512	104401	050345		TYPE	.BLNKS2		;TYPE 2 SPACES
64	022516	000761			BR	5\$		;LOOP
65	022520	001360		8\$:	BNE	5\$		;BRANCH IF NOT FINISHED
66	022522	104401	022530		TYPE	.67\$		;TYPE ASCII STRING
	022526	000401			BR	66\$		;GET OVER THE ASCII
				::67\$:	.ASCII	/)/		
	022532			66\$:				
67	022532	012037	022546	9\$:	MOV	(R0)+,10\$		;PICK UP DATA HEADER (DH) POINTER
68	022536	001404			BEQ	11\$		;BRANCH IF NONE
69	022540	104401	001231		TYPE	.\$CRLF		;CARRIAGE RETURN-LINE FEED
70	022544	104401			TYPE			
71	022546	000000		10\$:	.WORD	0		; 'DH' POINTER GOES HERE
72	022550	012001		11\$:	MOV	(R0)+,R1		;PICKUP DATA TABLE (DT) POINTER
73	022552	001450			BEQ	20\$		;BRANCH IF NONE
74	022554	005005			CLR	R5		;SET INDENT SWITCH
75	022556	012000			MOV	(R0)+,R0		;DATA FORMAT (DF) POINTER
76	022560	012002			MOV	(R0)+,R2		;NUMBER OF DH'S TO TYPE
77	022562	001441			BEQ	19\$		;BRANCH IF DH NUMBER IS 0
78	022564	005105			COM	R5		;NO INDENT
79	022566	104401	001231		TYPE	.\$CRLF		;CARRIAGE RETURN-LINE FEED
80	022572	112003		12\$:	MOVB	(R0)+,R3		;NUMBER OF DATA WORDS TO TYPE
81	022574	112004			MOVB	(R0)+,R4		;AND HOW TO TYPE THEM
82	022576	006004		13\$:	ROR	R4		;OCTAL OR DECIMAL?
83	022600	103403			BCS	14\$		;DECIMAL--BRANCH
84	022602	013146			MOV	@(R1)+,-(SP)		;SAVE @(R1)+ FOR TYPEOUT
	022604	104402			TYPOC			;GO TYPE--OCTAL ASCII(ALL DIGITS)
85	022606	000402			BR	15\$		
86	022610			14\$:				
	022610	013146			MOV	@(R1)+,-(SP)		;SAVE @(R1)+ FOR TYPEOUT
	022612	104405			TYPDS			;GO TYPE--DECIMAL ASCII WITH SIGN
87	022614	005303		15\$:	DEC	R3		;MORE NUMBERS TO TYPE?
88	022616	001403			BEQ	16\$		;NO--BRANCH
89	022620	104401	050345		TYPE	.BLNKS2		;TYPE 2 SPACES
90	022624	000764			BR	13\$		;LOOP
91	022626	005302		16\$:	DEC	R2		;MORE DH'S?
92	022630	003421			BLE	20\$		;NO--BRANCH
93	022632	104401	001231		TYPE	.\$CRLF		;YES--START A NEW LINE
94	022636	005105			COM	R5		;INDENT?
95	022640	001002			BNE	17\$		;NO--BRANCH
96	022642	104401	050345		TYPE	.BLNKS2		;TYPE 2 SPACES
97	022646	012037	022654	17\$:	MOV	(R0)+,18\$		;GET NEXT DH
98	022652	104401			TYPE			;AND TYPE IT
99	022654	000000		18\$:	.WORD	0		;DH POINTER GOES HERE
100	022656	104401	001231		TYPE	.\$CRLF		;CARRIAGE RETURN-LINE FEED
101	022662	005705			TST	R5		;INDENT?
102	022664	001342			BNE	12\$		;NO--BRANCH
103	022666	104401	050345	19\$:	TYPE	.BLNKS2		;TYPE 2 SPACES
104	022672	000737			BR	12\$		;LOOP
105	022674	104413		20\$:	RESREG			;RESTORE R0 - R5
106	022676	000207			RTS	PC		;RETURN
107	022700	022710	022772	023024	PFECH:	PFECH1,PFECH2,PFECH3,PFECH4		;WORDS DEFINING TABLES BELOW
108	022710	120	117	127	PFECH1:	.ASCII ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?		
109	022772	124	105	123	PFECH2:	.ASCII ?TESTNO ERR PC CPUERREG?		

10.

1

.SBT/L TYPE ROUTINE

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

*CALL:
 *1) USING A TRAP INSTRUCTION
 * TYPE ,MESADR ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 *OR
 * TYPE
 * MESADR
 *

023042	105737	001173	\$TYPE:	TSTB	\$TPFLG	::IS THERE A TERMINAL?
023046	100002			BPL	1\$	::BR IF YES
023050	000000			HALT		::HALT HERE IF NO TERMINAL
023052	000430			BR	3\$	::LEAVE
023054	010046		1\$:	MOV	R0,-(SP)	::SAVE R0
023056	017600	000002		MOV	@2(SP),R0	::GET ADDRESS OF ASCIZ STRING
023062	122737	000001 001254		CMPS	#APTENV,\$ENV	::RUNNING IN APT MODE
023070	001011			BNE	62\$	::NO,GO CHECK FOR APT CONSOLE
023072	132737	000100 001255		BITB	#APTSPOOL,\$ENV	::SPOOL MESSAGE TO APT
023100	001405			BEQ	62\$	::NO,GO CHECK FOR CONSOLE
023102	010037	023112		MOV	R0,61\$	::SETUP MESSAGE ADDRESS FOR APT
023106	004737	026756		JSR	PC,\$ATY3	::SPOOL MESSAGE TO APT
023112	000000		61\$:	.WORD	0	::MESSAGE ADDRESS
023114	132737	000040 001255	62\$:	BITB	#APTCSP, \$ENV	::APT CONSOLE SUPPRESSED
023122	001003			BNE	60\$	::YES,SKIP TYPE OUT
023124	112046		2\$:	MOVB	(R0)+,-(SP)	::PUSH CHARACTER TO BE TYPED ONTO STACK
023126	001005			BNE	4\$	::BR IF IT ISN'T THE TERMINATOR
023130	005726			TST	(SP)+	::IF TERMINATOR POP IT OFF THE STACK
023132	012600		60\$:	MOV	(SP)+,R0	::RESTORE R0
023134	062716	000002	3\$:	ADD	#2,(SP)	::ADJUST RETURN PC
023140	000002			RTI		::RETURN
023142	122716	000011	4\$:	CMPS	#HT,(SP)	::BRANCH IF <HT>
023146	001430			BEQ	8\$	
023150	122716	000200		CMPS	#CRLF,(SP)	::BRANCH IF NOT <CRLF>
023154	001006			BNE	5\$	
023156	005726			TST	(SP)+	::POP <CR><LF> EQUIV
023160	104401			TYPE		::TYPE A CR AND LF
023162	001231			\$CRLF		
023164	105037	023372		CLRB	\$CHARCNT	::CLEAR CHARACTER COUNT
023170	000755			BR	2\$	::GET NEXT CHARACTER
023172	004737	023254	5\$:	JSR	PC,\$TYPEC	::GO TYPE THIS CHARACTER
023176	123726	001172	6\$:	CMPS	\$FILLC,(SP)+	::IS IT TIME FOR FILLER CHARS.?
023202	001350			BNE	2\$	::IF NO GO GET NEXT CHAR.
023204	013746	001170		MOV	\$NULL,-(SP)	::GET # OF FILLER CHARS. NEEDED
						::AND THE NULL CHAR.
023210	105366	000001	7\$:	DECB	1(SP)	::DOES A NULL NEED TO BE TYPED?
023214	002770			BLT	6\$	::BR IF NO--GO POP THE NULL OFF OF STACK
023216	004737	023254		JSR	PC,\$TYPEC	::GO TYPE A NULL
023222	105337	023372		DECB	\$CHARCNT	::DO NOT COUNT AS A COUNT
023226	000770			BR	7\$	::LOOP

:HORIZONTAL TAB PROCESSOR

023230	112716	000040		8\$:	MOVB	#', (SP)	::REPLACE TAB WITH SPACE
023234	004737	023254		9\$:	JSR	PC, \$TYPEC	::TYPE A SPACE
023240	132737	000007	023372		BITB	#7, \$CHARCNT	::BRANCH IF NOT AT
023246	001372				BNE	9\$	::TAB STOP
023250	005726				TST	(SP)+	::POP SPACE OFF STACK
023252	000724				BR	2\$	::GET NEXT CHARACTER
023254				\$TYPEC:			
023254	105777	155700			TSTB	@\$TKS	::CHAR IN KYBD BUFFER?
023260	100022				BPL	10\$	::BR IF NOT
023262	017746	155674			MOV	@\$TKB, -(SP)	::GET CHAR
023266	042716	177600			BIC	#177600, (SP)	::STRIP EXTRANEIOUS BITS
023272	122716	000023			CMPS	#\$XOFF, (SP)	::WAS CHAR XOFF
023276	001012				BNE	102\$	::BR IF NOT
023300				101\$:			
023300	105777	155654			TSTB	@\$TKS	::WAIT FOR CHAR
023304	100375				BPL	101\$	
023306	117716	155650			MOVB	@\$TKB, (SP)	::GET CHAR
023312	042716	177600			BIC	#177600, (SP)	::STRIP IT
023316	122716	000021			CMPS	#\$XON, (SP)	::WAS IT XON?
023322	001366				BNE	101\$	::BR IF NOT
023324				102\$:			
023324	005726				TST	(SP)+	::FIX STACK
023326				10\$:			
023326	105777	155632			TSTB	@\$TPS	::WAIT UNTIL PRINTER IS READY
023332	100375				BPL	10\$	
023334	116677	000002	155624		MOVB	2(SP), @\$TPB	::LOAD CHAR TO BE TYPED INTO DATA REG.
023342	122766	000015	000002		CMPS	#CR, 2(SP)	::IS CHARACTER A CARRIAGE RETURN?
023350	001003				BNE	1\$	::BRANCH IF NO
023352	105037	023372			CLRB	\$CHARCNT	::YES--CLEAR CHARACTER COUNT
023356	000406				BR	\$TYPEX	::EXIT
023360	122766	000012	000002	1\$:	CMPS	#LF, 2(SP)	::IS CHARACTER A LINE FEED?
023366	001402				BEQ	\$TYPEX	::BRANCH IF YES
023370	105227				INCB	(PC)+	::COUNT THE CHARACTER
023372	000000			\$CHARCNT: .WORD	0		::CHARACTER COUNT STORAGE
023374	000207			\$TYPEX: RTS	PC		

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*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

```

023376	017646	000000		STYPOS:	MOV	@(SP),-(SP)	;;PICKUP THE MODE
023402	116637	000001	023621		MOVB	1(SP),%0FILL	;;LOAD ZERO FILL SWITCH
023410	112637	023623			MOVB	(SP)+,%MODE+1	;;NUMBER OF DIGITS TO TYPE
023414	062716	000002			ADD	#2,(SP)	;;ADJUST RETURN ADDRESS
023420	000406				BR	STYPON	
023422	112737	000001	023621	STYPOC:	MOVB	#1,%0FILL	;;SET THE ZERO FILL SWITCH
023430	112737	000006	023623		MOVB	#6,%MODE+1	;;SET FOR SIX(6) DIGITS
023436	112737	000005	023620	STYPON:	MOVB	#5,%CNT	;;SET THE ITERATION COUNT
023444	010346				MOV	R3,-(SP)	;;SAVE R3
023446	010446				MOV	R4,-(SP)	;;SAVE R4
023450	010546				MOV	R5,-(SP)	;;SAVE R5
023452	113704	023623			MOVB	%MODE+1,R4	;;GET THE NUMBER OF DIGITS TO TYPE
023456	005404				NEG	R4	
023460	062704	000006			ADD	#6,R4	;;SUBTRACT IT FOR MAX. ALLOWED
023464	110437	023622			MOVB	R4,%MODE	;;SAVE IT FOR USE
023470	113704	023621			MOVB	%0FILL,R4	;;GET THE ZERO FILL SWITCH
023474	016605	000012			MOV	12(SP),R5	;;PICKUP THE INPUT NUMBER
023500	005003				CLR	R3	;;CLEAR THE OUTPUT WORD
023502	006105			1\$:	ROL	R5	;;ROTATE MCB INTO 'C'
023504	000404				BR	3\$	;;GO DO MSB
023506	006105			2\$:	ROL	R5	;;FORM THIS DIGIT
023510	006105				ROL	R5	
023512	006105				ROL	R5	
023514	010503				MOV	R5,R3	
023516	006103			3\$:	ROL	R3	;;GET LSB OF THIS DIGIT
023520	105337	023622			DECB	%MODE	;;TYPE THIS DIGIT?
023524	100016				BPL	7\$	;;BR IF NO
023526	042703	177770			BIC	#177770,R3	;;GET RID OF JUNK
023532	001002				BNE	4\$	;;TEST FOR 0
023534	005704				TST	R4	;;SUPPRESS THIS 0?
023536	001403				BEQ	5\$	;;BR IF YES
023540	005204			4\$:	INC	R4	;;DON'T SUPPRESS ANYMORE 0'S

023542	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
023546	052703	000040	5\$:	BIS	#' ,R3	::MAKE ASCII IF NOT ALREADY
023552	110337	023616		MOVB	R3,8\$	::SAVE FOR TYPING
023556	104401	023616		TYPE	8\$	::GO TYPE THIS DIGIT
023562	105337	023620	7\$:	DECB	\$OCNT	::COUNT BY 1
023566	003347			BGT	2\$	::BR IF MORE TO DO
023570	002402			BLT	6\$	::BR IF DONE
023572	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
023574	000744			BR	2\$	::GO DO THE LAST DIGIT
023576	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
023600	012604			MOV	(SP)+,R4	::RESTORE R4
023602	012603			MOV	(SP)+,R3	::RESTORE R3
023604	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
023612	012616			MOV	(SP)+,(SP)	
023614	000002			RTI		::RETURN
023616	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
023617	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
023620	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
023621	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
023622	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

.SB*TL 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

023624				\$TYPDS:	MOV	R0,-(SP)	;;PUSH R0 ON STACK
023624	010046				MOV	R1,-(SP)	;;PUSH R1 ON STACK
023626	010146				MOV	R2,-(SP)	;;PUSH R2 ON STACK
023630	010246				MOV	R3,-(SP)	;;PUSH R3 ON STACK
023632	010346				MOV	R5,-(SP)	;;PUSH R5 ON STACK
023634	010546				MOV	#20200,-(SP)	;;SET BLANK SWITCH AND SIGN
023636	012746	020200			MOV	20(SP),R5	;;GET THE INPUT NUMBER
023642	016605	000020			BPL	1\$	;;BR IF INPUT IS POS.
023646	100004				NEG	R5	;;MAKE THE BINARY NUMBER POS.
023650	005405				MOVB	#'-,1(SP)	;;MAKE THE ASCII NUMBER NEG.
023652	112766	000055	000001	1\$:	CLR	R0	;;ZERO THE CONSTANTS INDEX
023660	005000				MOV	#\$DBLK,R3	;;SETUP THE OUTPUT POINTER
023662	012703	024040			MOVB	#',(R3)+	;;SET THE FIRST CHARACTER TO A BLANK
023666	112723	000040		2\$:	CLR	R2	;;CLEAR THE BCD NUMBER
023672	005002				MOV	\$DTBL(R0),R1	;;GET THE CONSTANT
023674	016001	024030		3\$:	SUB	R1,R5	;;FORM THIS BCD DIGIT
023700	160105				BLT	4\$	;;BR IF DONE
023702	002402				INC	R2	;;INCREASE THE BCD DIGIT BY 1
023704	005202				BR	3\$	
023706	000774			4\$:	ADD	R1,R5	;;ADD BACK THE CONSTANT
023710	060105				TST	R2	;;CHECK IF BCD DIGIT-0
023712	005702				BNE	5\$	;;FALL THROUGH IF 0
023714	001002				TSTB	(SP)	;;STILL DOING LEADING 0'S?
023716	105716				BMI	7\$	;;BR IF YES
023720	100407				ASLB	(SP)	;;MSD?
023722	106316			5\$:	BCC	6\$	;;BR IF NO
023724	103003				MOVB	1(SP),-1(R3)	;;YES--SET THE SIGN
023726	116663	000001	177777	6\$:	BIS	#'0,R2	;;MAKE THE BCD DIGIT ASCII
023734	052702	000060		7\$:	BIS	#',R2	;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
023740	052702	000040			MOVB	R2,(R3)+	;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
023744	110223				TST	(R0)+	;;JUST INCREMENTING
023746	005720				CMP	R0,#10	;;CHECK THE TABLE INDEX
023750	020027	000010			BLT	2\$	;;GO DO THE NEXT DIGIT
023754	002746				BGT	8\$	;;GO TO EXIT
023756	003002				MOV	R5,R2	;;GET THE LSD
023760	010502				BR	6\$	;;GO CHANGE TO ASCII
023762	000764			8\$:	TSTB	(SP)+	;;WAS THE LSD THE FIRST NON-ZERO?
023764	105726				BPL	9\$	;;BR IF NO
023766	100003				MOVB	-1(SP),-2(R3)	;;YES--SET THE SIGN FOR TYPING
023770	116663	177777	177776	9\$:	CLRB	(R3)	;;SET THE TERMINATOR
023776	105013				MOV	(SP)+,R5	;;POP STACK INTO R5
024000	012605				MOV	(SP)+,R3	;;POP STACK INTO R3
024002	012603				MOV	(SP)+,R2	;;POP STACK INTO R2
024004	012602				MOV	(SP)+,R1	;;POP STACK INTO R1
024006	012601						

024010	012600			MOV	(SP)+,RC	::POP STACK INTO RO
024012	104401	024040		TYPE	,SDBLK	::NOW TYPE THE NUMBER
024016	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
024024	012616			MOV	(SP)+,(SP)	
024026	000002			RTI		::RETURN TO USER
024030	023420			\$DTBL:	10000.	
024032	001750				1000.	
024034	000144				100.	
024036	000012				10.	
024040				\$SDBLK:	.BLKW 4	

.SBTTL TTY INPUT ROUTINE

```

*****
ENABL  LSB
024050 000000 $TKCNT: .WORD 0      ;;NUMBER OF ITEMS IN QUEUE
024052 000000 $TKQIN: .WORD 0      ;;INPUT POINTER
024054 000000 $TKQOUT: .WORD 0      ;;OUTPUT POINTER
024056      $TKQSRV: .BLKB 1      ;;TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

;*TK INITIALIZE ROUTINE
;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
;CALL:
;JSR PC,$TKINT
;RETURN
024060 005037 024050 $TKINT: CLR $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
024064 012737 024056 024052 MOV # $TKQSRV,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
024072 013737 024052 024054 MOV $TKQIN,$TKQOUT ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
024100 012737 024130 000060 MOV # $TKSRV,@ $TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
024106 012737 000200 000062 MOV #200,@ $TKVEC+2 ;;'BR' LEVEL 4
024114 005777 155042 TST @ $TKB      ;;CLEAR DONE FLAG
024120 012777 000100 155032 MOV #100,@ $TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
024126 000207 RTS PC      ;;RETURN TO CALLER

;*TK SERVICE ROUTINE
;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
;*IT IN THE QUEUE.
;*IF THE CHARACTER IS A 'CONTROL-C' (^C) $TKINT IS CALLED AND
;*UPON RETURN EXIT IS MADE TO THE 'CONTROL-C' RESTART ADDRESS (SHUT)
024130 117746 155026 $TKSRV: MOVB @ $TKB,-(SP)      ;;PICKUP THE CHARACTER
024134 042716 177600 BIL #^C177,(SP)      ;;STRIP THE JUNK
024140 021627 000021 CMP (SP),#$XON      ;;IS IT A RANDOM XON?
024144 001002 BNE 30$      ;;BRANCH IF NO
024146 005726 TST (SP)+      ;;CLEAN RANDOM XON OFF STACK
024150 000002 RTI      ;;RETURN
024152 30$:
024152 021627 000003 CMP (SP),#3      ;;IS IT A CONTROL C?
024156 001007 BNE 1$      ;;BRANCH IF NO
024160 104401 025272 TYPE , $CNTLC      ;;TYPE A CONTROL-C (^C)
024164 004737 024060 JSR PC,$TKINT      ;;INIT THE KEYBOARD
024170 005726 TST (SP)+      ;;CLEAN UP STACK
024172 000137 025334 JMP SHUT      ;;CONTROL C RESTART
024176 021627 000007 1$: CMP (SP),#7      ;;IS IT A CONTROL G?
024202 001004 BNE 2$      ;;BRANCH IF NO
024204 022737 000176 001154 CMP #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
024212 001500 BEQ 6$      ;;GO TO SWR CHANGE
024214 2$:
024214 022737 000001 024050 CMP #1,$TKCNT      ;;IS THE QUEUE FULL?
024222 001004 BNE '      ;;BRANCH IF NO
024224 104401 001224 TYPE , $BELL      ;;RING THE TTY BELL

```

```

024230 005726      TST      (SP)+      ;;CLEAN CHARACTER OFF OF STACK
024232 000451      BR        5$          ;;EXIT
024234 021627 000023 3$:      CMP      (SP),#23      ;;IS IT A CONTROL-S?
024240 001021      BNE      32$          ;;BRANCH IF NO
024242 005077 154712      CLR      @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
024246 005726      TST      (SP)+      ;;CLEAN CHAR OFF STACK
024250 105177 154704 31$:    TSTB     @STKS      ;;WAIT FOR A CHAR
024254 100375      BPL      31$          ;;LOOP UNTIL ITS THERE
024256 117746 154700      MOVB     @STKB,-(SP)      ;;GET THE CHARACTER
024262 042716 177600      BIC      #'C177,(SP)      ;;MAKE IT 7-BIT ASCII
024266 022627 000021      CMP      (SP)+,#21      ;;IS IT A CONTROL-Q?
024272 001366      BNE      31$          ;;BRANCH IF NO
024274 012777 000100 154656  MOV      #100,@STKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
024302 000002      RTI                    ;;RETURN
024304 005237 024050 32$:    INC      $TKCNT      ;;COUNT THIS CHARACTER
024310 021627 000140      CMP      (SP),#140      ;;IS IT UPPER CASE?
024314 002405      BLT      4$          ;;BRANCH IF YES
024316 021627 000175      CMP      (SP),#175      ;;IS IT A SPECIAL CHAR?
024322 003002      BGT      4$          ;;BRANCH IF YES
024324 042716 000040      BIC      #40,(SP)      ;;MAKE IT UPPER CASE
024330 112677 177516 4$:    MOVB     (SP)+,@STKQIN      ;;AND PUT IT IN QUEUE
024334 005237 024052      INC      $TKQIN      ;;UPDATE THE POINTER
024340 023727 024052 024057  CMP      $TKQIN,#$TKQEND      ;;GO OFF THE END?
024346 001003      BNE      5$          ;;BRANCH IF NO
024350 012737 024056 024052  MOV      #$TKQSRT,$TKQIN      ;;RESET THE POINTER
024356 000002 5$:      RTI                    ;;RETURN

```

;;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
;;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
;;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
;;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.

```

024360 022737 000176 001154 $CKSWR: CMP      #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
024366 001124      BNE      15$          ;;EXIT IF NOT
024370 105777 154564      TSTB     @STKS      ;;IS A CHAR WAITING?
024374 100121      BPL      15$          ;;IF NOT, EXIT
024376 117746 154560      MOVB     @STKB,-(SP)      ;;YES
024402 042716 177600      BIC      #'C177,(SP)      ;;MAKE IT 7-BIT ASCII
024406 021627 000007      CMP      (SP),#7      ;;IS IT A CONTROL-G?
024412 001300      BNE      2$          ;;IF NOT, PUT IT IN THE TTY QUEUE
                                ;;AND EXIT

```

;;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
;;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
;;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.

```

024414 123727 001150 000001 6$:    CMPEB    $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
024422 001674      BEQ      2$          ;;BRANCH IF YES
024424 005726      TST      (SP)+      ;;CLEAR CONTROL-G OFF STACK
024426 004737 024060      JSR      PC,$TKINT      ;;FLUSH THE TTY INPUT QUEUE
024432 005077 154522      CLR      @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
024436 112737 000001 001151  MOVB     #1,$INTAG      ;;SET INTERRUPT MODE INDICATOR

024444 104401 025304      TYPE      ,$CNTLG      ;;ECHO THE CONTROL-G (^G)
024450 104401 025311      $GTSWR: TYPE      ,$MSWR      ;;TYPE CURRENT CONTENTS
024454 013746 000176      MOV      SWREG,-(SP)      ;;SAVE SWREG FOR TYPEOUT
024460 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)

```

024462	104401	025322		TYPE	,\$MNEW	::PROMPT FOR NEW SWR
024466	005046		19\$:	CLR	-(SP)	::CLEAR COUNTER
024470	005046			CLR	-(SP)	::THE NEW SWR
024472	105777	154462	7\$:	TSTB	@\$TKS	::CHAR THERE?
024476	100375			BPL	7\$	::IF NOT TRY AGAIN
024500	117746	154456		MOVB	@\$TKB, -(SP)	::PICK UP CHAR
024504	042716	177600		BIC	#^C177, (SP)	::MAKE IT 7-BIT ASCII
024510	021627	000003		CMP	(SP), #3	::IS IT A CONTROL-C?
024514	001015			BNE	9\$	::BRANCH IF NOT
024516	104401	025272		TYPE	,\$CNTLC	::YES, ECHO CONTROL-C (^C)
024522	062706	000006		ADD	#6, SP	::CLEAN UP STACK
024526	123727	001151	000001	CMPB	\$INTAG, #1	::REENABLE TTY KEYBOARD INTERRUPTS?
024534	001003			BNE	8\$	::BRANCH IF NO
024536	012777	000100	154414	MOV	#100, @\$TKS	::ALLOW TTY KEYBOARD INTERRUPTS
024544	000137	025334	8\$:	JMP	SHUT	::CONTROL-C RESTART
024550	021627	000025		CMP	(SP), #25	::IS IT A CONTROL-U?
024554	001005		9\$:	BNE	10\$	::BRANCH IF NOT
024556	104401	025277		TYPE	,\$CNTLU	::YES, ECHO CONTROL-U (^U)
024562	062706	000006	20\$:	ADD	#6, SP	::IGNORE PREVIOUS INPUT
024566	000737			BR	19\$	::LET'S TRY IT AGAIN
024570	021627	000015		CMP	(SP), #15	::IS IT A <CR>?
024574	001022		10\$:	BNE	16\$	::BRANCH IF NO
024576	005766	000004		TST	4(SP)	::YES, IS IT THE FIRST CHAR?
024602	001403			BEQ	11\$	::BRANCH IF YES
024604	016677	000002	154342	MOV	2(SP), @SWR	::SAVE NEW SWR
024612	062706	000006		ADD	#6, SP	::CLEAN UP STACK
024616	104401	001231		TYPE	,\$CRLF	::ECHO <CR> AND <LF>
024622	123727	001151	000001	CMPB	\$INTAG, #1	::RE-ENABLE TTY KBD INTERRUPTS?
024630	001003			BNE	15\$	::BRANCH IF NOT
024632	012777	000100	154320	MOV	#100, @\$TKS	::RE-ENABLE TTY KBD INTERRUPTS
024640	000002		15\$:	RTI		::RETURN
024642	004737	023254	16\$:	JSR	PC, \$TYPEC	::ECHO CHAR
024646	021627	000060		CMP	(SP), #60	::CHAR < 0?
024652	002420			BLT	18\$	::BRANCH IF YES
024654	021627	000067		CMP	(SP), #67	::CHAR > 7?
024660	003015			BGT	18\$	::BRANCH IF YES
024662	042726	000060		BIC	#60, (SP)+	::STRIP-OFF ASCII
024666	005766	000002		TST	2(SP)	::IS THIS THE FIRST CHAR
024672	001403			BEQ	17\$	::BRANCH IF YES
024674	006316			ASL	(SP)	::NO, SHIFT PRESENT
024676	006316			ASL	(SP)	::CHAR OVER TO MAKE
024700	006316			ASL	(SP)	::ROOM FOR NEW ONE.
024702	005266	000002	17\$:	INC	2(SP)	::KEEP COUNT OF CHAR
024706	056616	177776		BIS	-2(SP), (SP)	::SET IN NEW CHAR
024712	000667			BR	7\$	::GET THE NEXT ONE
024714	104401	001230	18\$:	TYPE	,\$QUES	::TYPE ?<CR><LF>
024720	000720			BR	20\$	::SIMULATE CONTROL-U
			.DSABL	LSB		

::*****

```

: *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
: *CALL:
: *      RDCHR
: *      RETURN HERE
: *      :;GET A CHARACTER FROM THE QUEUE
: *      :;CHARACTER IS ON THE STACK
: *      :;WITH PARITY BIT STRIPPED OFF

024722 011646          $RDCHR: MOV      (SP),-(SP)      :;PUSH DOWN THE PC AND
024724 016666 000004 000002      MOV      4(SP),2(SP)    :;THE PS
024732 005066 000004          CLR      4(SP)      :;GET READY FOR A CHARACTER
024736 005046          CLR      -(SP)      :;PUT NEW PS ON STACK
024740 012746 024746          MOV      #64$,-(SP)    :;PUT NEW PC ON STACK
024744 000002          RTI          :;POP NEW PC AND PS
024746          64$:
024746 005737 024050      1$:      TST      $TKCNT      :;WAIT ON A CHARACTER
024752 001775          BEQ      1$
024754 005337 024050      DEC      $TKCNT      :;DECREMENT THE COUNTER
024760 117766 177070 000004      MOV      @STKQOUT,4(SP) :;GET ONE CHARACTER
024766 005237 024054      INC      $TKJOUT      :;UPDATE THE POINTER
024772 023727 024054 024057      CMP      $TKQOUT,$TKQEND :;DID IT GO OFF OF THE END?
025000 001003          BNE      2$      :;BRANCH IF NO
025002 012737 024056 024054      MOV      #STKQSRST,$TKQOUT :;RESET THE POINTER
025010 000002          RTI          :;RETURN
: *****
: *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
: *CALL:
: *      RDLIN
: *      RETURN HERE
: *      :;INPUT A STRING FROM THE TTY
: *      :;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
: *      :;TERMINATOR WILL BE A BYTE OF ALL 0'S

025012 010346          $RDLIN: MOV      R3, -(SP)      :;SAVE R3
025014 005046          CLR      -(SP)      :;CLEAR THE RUBOUT KEY
025016 012703 025246      1$:      MOV      #STTYIN,R3    :;GET ADDRESS
025022 022703 025272      2$:      CMP      #STTYIN+20.,R3 :;BUFFER FULL?
025026 101456          BLOS      4$      :;BR IF YES
025030 104410          RDCHR      :;GO READ ONE CHARACTER FROM THE TTY
025032 112613          MOV      (SP)+,(R3)    :;GET CHARACTER
025034 122713 000177      10$:     CMP      #177,(R3)    :;IS IT A RUBOUT
025040 001022          BNE      5$      :;BR IF NO
025042 005716          TST      (SP)      :;IS THIS THE FIRST RUBOUT?
025044 001007          BNE      6$      :;BR IF NO
025046 112737 000134 025244      MOV      #'\\,9$      :;TYPE A BACK SLASH
025054 104401 025244          TYPE      ,9$
025060 012716 177777      MOV      #-1,(SP)    :;SET THE RUBOUT KEY
025064 005303          DEC      R3          :;BACKUP BY ONE
025066 020327 025246      6$:      CMP      R3,#STTYIN    :;STACK EMPTY?
025072 103434          BLO      4$      :;BR IF YES
025074 111337 025244      MOV      (R3),9$      :;SETUP TO TYPEOUT THE DELETED CHAR.
025100 104401 025244          TYPE      ,9$
025104 000746          BR      2$      :;GO TYPE
025106 005716          5$:      TST      (SP)      :;GO READ ANOTHER CHAR.
025110 001406          BEQ      7$      :;RUBOUT KEY SET?
025112 112737 000134 025244      MOV      #'\\,9$      :;BR IF NO
025120 104401 025244          TYPE      ,9$      :;TYPE A BACK SLASH
025124 005016          CLR      (SP)      :;CLEAR THE RUBOUT KEY
025126 122713 000025      7$:      CMP      #25,(R3)    :;IS CHARACTER A CTRL U?
025132 001003          BNE      8$      :;BR IF NO

```

025134	104401	025277		TYPE	,SCNTLU	::TYPE A CONTROL 'U'
025140	000726			BR	1\$	::GO START OVER
025142	122713	000022	8\$:	CMPB	#22,(R3)	::IS CHARACTER A '^R'?
025146	001011			BNE	3\$	::BRANCH IF NO
025150	105013			CLRB	(R3)	::CLEAR THE CHARACTER
025152	104401	001231		TYPE	,SCRLF	::TYPE A 'CR' & 'LF'
025156	104401	025246		TYPE	,STTYIN	::TYPE THE INPUT STRING
025162	000717			BR	2\$	::GO PICKUP ANOTHER CHACTER
025164	104401	001230	4\$:	TYPE	,SQUES	::TYPE A '?'
025170	000712			BR	1\$	::CLEAR THE BUFFER AND LOOP
025172	111337	025244	3\$:	MOVB	(R3),9\$	::ECHO THE CHARACTER
025176	104401	025244		TYPE	,9\$	
025202	122723	000015		CMPB	#15,(R3)+	::CHECK FOR RETURN
025206	001305			BNE	2\$	::LOOP IF NOT RETURN
025210	105063	177777		CLRB	-1(R3)	::CLEAR RETURN (THE 15)
025214	104401	001232		TYPE	,SLF	::TYPE A LINE FEED
025220	005726			TST	(SP)+	::CLEAN RUBOUT KEY FROM THE STACK
025222	012603			MOV	(SP)+,R3	::RESTORE R3
025224	011646			MOV	(SP),-(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE
025226	016666	000004	000002	MOV	4(SP),2(SP)	::FIRST ASCII CHARACTER ON IT
025234	012766	025246	000004	MOV	#STTYIN,4(SP)	
025242	000002			RTI		::RETURN
025244	000		9\$:	.BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE
025245	000			.BYTE	0	::TERMINATOR
025246				STTYIN:	.BLKB 20.	::RESERVE 20. BYTES FOR TTY INPUT
025272	136	103	015	SCNTLC:	.ASCIIZ /^C/<15><12>	::CONTROL 'C'
025277	136	125	015	SCNTLU:	.ASCIIZ /^U/<15><12>	::CONTROL 'U'
025304	136	107	015	SCNTLG:	.ASCIIZ /^G/<15><12>	::CONTROL 'G'
025311	015	012	123	SMSWR:	.ASCIIZ <15><12>/SWR = /	
025322	040	040	116	SMNEW:	.ASCIIZ / NEW = /	
				.EVEN		
2						
3	025334	005737	000042	SHUT:	TST	@#42
4	025340	001002			BNE	1\$
5	025342	000137	004712		JMP	START2
6	025346	005037	001330	1\$:	CIR	DRVSEL
7	025352	000137	021334		JMP	\$EOP
						::ANY MONITOR PRESENT ?
						::BR IF YES
						::GO TO 'START2'
						::FUDGE NO DRIVES SELECTED
						::RETURN CONTROL TO MONITOR

1

.SBTTL SCOPE HANDLER ROUTINE

```

*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1      LOOP ON TEST
*SW11=1      INHIBIT ITERATIONS
*SW09=1      LOOP ON ERROR
*CALL
*          SCOPE          ;;SCOPE=IOT

```

```

025356          $SCOPE:
025356 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
025360 032777 040000 153566 1$:      BIT      #BIT14,@SWR          ;;LOOP ON PRESENT TEST?
025366 001402          BEQ      9$          ;;NO IF SW14=0.
025370 000137 025740          JMP      $OVER          ;;JUMP OVER SCOPE ROUTINE
025374          9$:
          ;;*****START OF CODE FOR THE XOR TESTER*****
025374 000416          $XTSTR: BR      6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
          ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
025376 013746 000004          MOV      @ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
025402 012737 025422 000004          MOV      #5$,@ERRVEC          ;;SET FOR TIMEOUT
025410 005737 177060          TST      @177060          ;;TIME OUT ON XOR?
025414 012637 000004          MOV      (SP)+,@ERRVEC          ;;RESTORE THE ERROR VECTOR
025420 000531          BR      $SVLAD          ;;GO TO THE NEXT TEST
025422 022626          5$:      CMP      (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
025424 012637 000004          MOV      (SP)+,@ERRVEC          ;;RESTORE THE ERROR VECTOR
025430 000474          BR      7$          ;;LOOP ON THE PRESENT TEST
025432          6$:;*****END OF CODE FOR THE XOR TESTER*****
025432 105737 001117          2$:      TSTB     $ERFLG          ;;HAS AN ERROR OCCURRED?
025436 001502          BEQ      3$          ;;BR IF NO
025440 022737 177777 022256          CMP      #-1,CPSAVE          ;;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED
025446 001455          BEQ      2003$          ;;KICK AROUND ROUTINE IF SO
025450 013746 000004          MOV      ERRVEC,-(SP)          ;;SAVE CONTENTS OF ERROR VECTOR
025454 012737 025472 000004          MOV      #2000$,ERRVEC          ;;SETUP 'TRAP' RETURN ADDRESS
025462 013737 177766 022256          MOV      177766,CPSAVE          ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
025470 000406          BR      2001$
025472 012737 177777 022256 2000$:    MOV      #-1,CPSAVE          ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
025500 012716 025506          MOV      #2001$,(SP)          ;;SETUP RETURN ADDRESS
025504 000002          RTI
025506 012637 000004          2001$:    MOV      (SP)+,ERRVEC          ;;RESTORE CONTENTS OF ERROR VECTOR

025512 022737 177777 022256 2002$:    CMP      #-1,CPSAVE          ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
025520 001430          BEQ      2003$          ;;BRANCH IF SO
025522 032737 000001 022256          BIT      #BIT00,CPSAVE          ;;SEE IF THE POWER MONITOR BIT IS ON
025530 001424          BEQ      2003$          ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
025532 042737 000001 177766          BIC      #BIT00,177766          ;;CLEAR THE BIT FOUND TO BE SET
025540 013746 001154          MOV      SWR,-(SP)          ;;SAVE SWR ADDRESS
025544 017646 000000          MOV      @$(SP),-(SP)          ;;SAVE SWR VALUE
025550 012737 000176 001154          MOV      #176,SWR          ;;GET SOFTWARE SWR ADDRESS
025556 011677 153372          MOV      (SP),@SWR          ;;GET CURRENT SWR VALUE
025562 042777 001000 153364          BIC      #BIT09,@SWR          ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
025570 104177          EMT      177          ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
025572 012676 000000          MOV      (SP)+,@$(SP)          ;;RESTORE SWR TO ORIGINAL VALUE
025576 012637 001154          MOV      (SP)+,SWR          ;;RESTORE SWR ADDRESS

```

SCOPE HANDLER ROUTINE

025602				2003\$:			
025602	123737	001131	001117		CMPB	SERMAX,SERFLG	::MAX. ERRORS FOR THIS TEST OCCURRED?
025610	101015				BHI	3\$	::BR IF NO
025612	032777	001000	153334		BIT	#BIT09,@SWR	::LOOP ON ERROR?
025620	001404				BEC	4\$	::BR IF NO
025622	013737	001124	001122	7\$:	MOV	\$LPERR,\$LPADR	::SET LOOP ADDRESS TO LAST SCOPE
025630	000443				BR	\$OVER	
025632	105037	001117		4\$:	CLRB	SERFLG	::ZERO THE ERROR FLAG
025636	005037	001220			CLR	\$TIMES	::(CLEAR THE NUMBER OF ITERATIONS TO MAKE
025642	000412				BR	1\$	::ESCAPE TO THE NEXT TEST
025644	032777	004000	153302	3\$:	BIT	#BIT11,@SWR	::INHIBIT ITERATIONS?
025652	001006				BNE	1\$	::BR IF YES
025654	005237	001120			INC	\$ICNT	::INCREMENT ITERATION COUNT
025660	023737	001220	001120		CMP	\$TIMES,\$ICNT	::CHECK THE NUMBER OF ITERATIONS MADE
025666	002024				BGE	\$OVER	::BR IF MORE ITERATION REQUIRED
025670	012737	000001	001120	1\$:	MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER
025676	013737	025754	001220		MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO
025704	105237	001116		\$SVLAD:	INCB	\$TSTNM	::COUNT TEST NUMBERS
025710	113737	001116	001240		MOVB	\$TSTNM,\$TESTN	::SET TEST NUMBER IN APT MAILBOX
025716	011637	001122			MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS
025722	011637	001124			MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS
025726	005037	001222			CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS
025732	112737	000001	001131		MOVB	#1,SERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
025740	013777	001116	153210	\$OVER:	MOV	\$TSTNM,@DISPLAY	::DISPLAY TEST NUMBER
025746	013716	001122			MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
025752	000002				RTI		::FIXES PS
025754	000001			\$MXCNT:	1		::MAX. NUMBER OF ITERATIONS

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

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

025756			\$SAVREG:	MOV	R0,-(SP)	::PUSH R0 ON STACK
025756	010046			MOV	R1,-(SP)	::PUSH R1 ON STACK
025760	010146			MOV	R2,-(SP)	::PUSH R2 ON STACK
025762	010246			MOV	R3,-(SP)	::PUSH R3 ON STACK
025764	010346			MOV	R4,-(SP)	::PUSH R4 ON STACK
025766	010446			MOV	R5,-(SP)	::PUSH R5 ON STACK
025770	010546			MOV	22(SP),-(SP)	::SAVE PS OF MAIN FLOW
025772	016646	000022		MOV	22(SP),-(SP)	::SAVE PC OF MAIN FLOW
025776	016646	000022		MOV	22(SP),-(SP)	::SAVE PS OF CALL
026002	016646	000022		MOV	22(SP),-(SP)	::SAVE PC OF CALL
026006	016646	000022		RTI		
026012	000002					
: *RESTORE R0-R5						
: *CALL:						
: * RESREG						
026014			\$RESREG:	MOV	(SP)+,22(SP)	::RESTORE PC OF CALL
026014	012666	000022		MOV	(SP)+,22(SP)	::RESTORE PS OF CALL
026020	012666	000022		MOV	(SP)+,22(SP)	::RESTORE PC OF MAIN FLOW
026024	012666	000022		MOV	(SP)+,22(SP)	::RESTORE PS OF MAIN FLOW
026030	012666	000022		MOV	(SP)+,R5	::POP STACK INTO R5
026034	012605			MOV	(SP)+,R4	::POP STACK INTO R4
026036	012604			MOV	(SP)+,R3	::POP STACK INTO R3
026040	012603			MOV	(SP)+,R2	::POP STACK INTO R2
026042	012602			MOV	(SP)+,R1	::POP STACK INTO R1
026044	012601			MOV	(SP)+,R0	::POP STACK INTO R0
026046	012600			RTI		
026050	000002					

.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.

026052	010046		\$TRAP: MOV	R0, -(SP)	:: SAVE R0
026054	016600	000002		MOV	2(SP), R0
026060	005740			TST	-(R0)
026062	111000			MOV	(R0), R0
026064	006300			ASL	R0
026066	016000	026106		MOV	\$TRPAD(R0), R0
026072	000200			RTS	R0

:: THIS IS USE TO HANDLE THE 'GETPRI' MACRO

026074	011646		\$TRAP2: MOV	(SP), -(SP)	:: MOVE THE PC DOWN
026076	016666	000004 000002		MOV	4(SP), 2(SP)
026104	000002			RTI	:: RESTORE THE PSW

.SBTTL TRAP TABLE

: THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 : BY THE 'TRAP' INSTRUCTION.

			:	ROUTINE	
			:	-----	
026106	026074		\$TRPAD: .WORD	\$TRAP2	
026110	023042			\$TYPE	:: CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
026112	023422			\$TYPOC	:: CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
026114	023376			\$TYPOS	:: CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
026116	023436			\$TYPON	:: CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
026120	023624			\$TYPDS	:: CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
026122	024450			\$GTSWR	:: CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
026124	024360			\$CKSWR	:: CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
026126	024722			\$RDCHR	:: CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
026130	025012			\$RDLIN	:: CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
026132	025756			\$SAVREG	:: CALL=SAVREG TRAP+12(104412) SAVE R0-R5 ROUTINE
026134	026014			\$RESREG	:: CALL=RESREG TRAP+13(104413) RESTORE R0-R5 ROUTINE

.SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCII ROUTINE

 *THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
 *UNSIGNED DECIMAL ASCII NUMBER.

*CALL
 * MOV NUMBER, -(SP) ;; PUT BINARY NUMBER ON THE STACK
 * JSR PC, @SSB2D ;; CALL
 * RETURN ;; ADDRESS OF THE 1ST ASCII CHAR. IS ON THE STACK

026136	016637	000002	026166	SSB2D:	MOV	2(SP), 1\$	:: SAVE BINARY NUMBER
026144	012746	026166			MOV	#1\$, -(SP)	:: SET POINTER
026150	004737	026172			JSR	PC, @SSB2D	:: CALL DOUBLE LENGTH CONVERT
026154	062716	000005			ADD	#5, (SP)	:: ONLY ALLOW FIVE CHARACTERS
026160	012666	000002			MOV	(SP)+, 2(SP)	:: PICKUP POINTER
026164	000207				RTS	PC	:: RETURN
026166	000000	000000		1\$:	.WORD	0,0	

.SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

 *THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
 *DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
 *POSITIVE.
 *CALL

* MOV #PNTR, -(SP) ;; POINTER TO LOW WORD OF BINARY NUMBER
 * JSR PC, @#\$DB2D
 * RETURN ;; THE FIRST ADDRESS OF ASCII
 ;; IS ON THE STACK

026172	104412		\$DB2D:	SAVREG	;; SAVE REGISTERS
026174	016602	000002		MOV 2(SP), R2	;; PICKUP THE DATA POINTER
026200	012700	026352		MOV #SDECVL, R0	;; GET ADDRESS OF 'SDECVL' STRING
026204	010066	000002		MOV R0, 2(SP)	;; PUT ADDRESS OF ASCII STRING ON STACK
026210	012201			MOV (R2)+, R1	;; PICKUP THE BINARY NUMBER
026212	012202			MOV (R2)+, R2	
026214	012737	000012	026270	MOV #10, 4\$	;; SET UP TO DO 10 CONVERSIONS
026222	012704	026302		MOV #STNPWR, R4	;; ADDRESS OF TEN POWER
026226	012705	026304		MOV #STNPWR+2, R5	
026232	005003		1\$:	CLR R3	;; CLEAR PARTIAL
026234	161401		2\$:	SUB (R4), R1	;; SUBTRACT TEN POWER
026236	005602			SBC R2	
026240	161502			SUB (R5), R2	
026242	002402			BLT 3\$	;; BR IF TEN POWER TO LARGE
026244	005203			INC R3	;; ADD 1 TO PARTIAL
026246	000772			BR 2\$	;; LOOP
026250	062401		3\$:	ADD (R4)+, R1	;; RESTORE SUBTRACTED VALUE
026252	005502			ADC R2	
026254	062402			ADD (R4)+, R2	
026256	022525			CMR (R5)+, (R5)+	;; MOVE TO NEXT TEN POWER
026260	052703	000060		BIS #0, R3	;; CHANGE PARTIAL TO ASCII
026264	110320			MOV R3, (R0)+	;; SAVE IT
026266	005327			DEC (PC)+	;; DONE?
026270	000000		4\$:	.WORD 0	
026272	001357			BNE 1\$	;; BR IF NO
026274	105020			CLRB (R0)+	;; TERMINATOR
026276	104413			RESREG	;; RESTORE REGISTERS
026300	000207			RTS PC	;; RETURN
026302	145000		STNPWR:	145000	;; 1.0E09
026304	035632			35632	
026306	160400			160400	;; 1.0E08
026310	002765			2765	
026312	113200			113200	;; 1.0E07
026314	000230			230	
026316	041100			041100	;; 1.0E06
026320	000017			17	
026322	103240			103240	;; 1.0E05
026324	000001			1	
026326	023420			23420	;; 1.0E04
026330	000000			0	
026332	001750			1750	;; 1.0E03
026334	000000			0	
026336	000144			144	;; 1.0E02
026340	000000			0	

026342 000012
026344 000000
026346 000001
026350 000000
026352

12
0
1
0
\$DECL: .BLKB 12.

::1.0E01
::1.0E00
::RESERVE STORAGE FOR ASCII STRING

.SBTTL TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS

 *THIS ROUTINE IS USED TO TYPE AN ASCII NUMBER SUPPRESSING THE
 *LEADING NUMBERS.
 *CALL
 * MOV #NUMADR,-(SP) ;;FIRST ADDRESS OF ASCII STRING
 * JSR PC,@#SSUPRS

026366	010046		SSUPRS:	MOV	R0,-(SP)	::SAVE R0
026370	016600	000004		MOV	4(SP),R0	::PICKUP THE POINTER
026374	105710		1\$:	TSTB	(R0)	::TERMINATOR?
026376	001403			BEQ	2\$	::BR IF YES
026400	122720	000060		CMPS	#'0,(R0)+	::IS THIS AN ASCII '0'?
026404	001773			BEQ	1\$	::BR IF YES
026406	005300		2\$:	DEC	R0	::BACKUP BY '1'
026410	010037	026416		MOV	R0,3\$	::SAVE FOR TYPING
026414	104401			TYPE		::GO TYPE
026416	000000		3\$:	.WORD	0	::ASCII POINTER GOES HERE
026420	012600			MOV	(SP)+,R0	::RESTORE R0
026422	012616			MOV	(SP)+,(SP)	::RESTORE THE STACK
026424	000207			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(+53)-1.
: CALL:

: JSR PC,\$RAND ;: CALL THE ROUTINE
: RETURN ;: RETURN HERE THE RANDOM
: ;: NUMBER WILL BE IN
: ;: \$HINUM,\$LONUM

026426
026426 010046
026430 010146
026432 010246
026434 013700 026526
026440 013701 026524
026444 012702 177771
026450 006300
026452 006101
026454 005202
026456 001374
026460 063700 026526
026464 005501
026466 063701 026524
026472 062700 001057
026476 005501
026500 062701 047401
026504 010037 026526
026510 010137 026524
026514 012602
026516 012601
026520 012600
026522 000207
026524 176543
026526 123456

\$RAND: MOV R0,-(SP) ;: PUSH R0 ON STACK
MOV R1,-(SP) ;: PUSH R1 ON STACK
MOV R2,-(SP) ;: PUSH R2 ON STACK
MOV \$LONUM,R0 ;: SET R0 WITH LOW
MOV \$HINUM,R1 ;: SET R1 WITH HIGH
MOV #-7,R2 ;: SET SHIFT COUNT
1\$: ASL R0 ;: SHIFT R0 LEFT AND
ROL R1 ;: ROTATE CARRY INTO R1 AND
INC R2 ;: CHECK FOR DONE
BNE 1\$;: CONTINUE SHIFT LOOP
ADD \$LONUM,R0 ;: ADD NUMBER TO MAKE X 129
ADC R1 ;: PROPOGATE CARRY
ADD \$HINUM,R1 ;: ADD NUMBER TO MAKE X 129
ADD #1057,R0 ;: ADD LOW CONSTANT
ADC R1 ;: PROPOGATE CARRY
ADD #47401,R1 ;: ADD HIGH CONSTANT
MOV R0,\$LONUM ;: SAVE R0
MOV R1,\$HINUM ;: SAVE R1
MOV (SP)+,R2 ;: POP STACK INTO R2
MOV (SP)+,R1 ;: POP STACK INTO R1
MOV (SP)+,R0 ;: POP STACK INTO R0
RTS PC ;: RETURN
\$HINUM: .WORD 176543
\$LONUM: .WORD 123456

.SBTTL INTEGER DIVIDE ROUTINE

```

*****
*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
*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

026530				\$DIV:	TRAP	;;PUSH OLD PSW AND PC ON STACK
026530	104400				BIC	;;STRIP AWAY CONDITION CODES
026532	042716	000017			MOV	;;PUSH R0 ON STACK
026536	010046				MOV	;;PUSH R1 ON STACK
026540	010146				MOV	;;PUSH R2 ON STACK
026542	010246				MOV	;;PUSH R3 ON STACK
026544	010346				MOV	;;SAVE A PLACE FOR SIGNS
026546	005046				CLR	;;SETUP THE ITERATION COUNTER
026550	012746	000021			MOV	;;PICKUP THE DIVIDEND
026554	016601	000024			MOV	
026560	016600	000022			MOV	
026564	100005				BPL	;;CHECK THE SIGN
026566	105366	000003			DECB	;;KEEP TRACK OF THE SIGN
026572	005400				NEG	;;AND NEGATE THE ORIGINAL
026574	005401				NEG	;;NUMBER
026576	005600				SBC	
026600	016602	000020		1\$:	MOV	;;PICKUP THE DIVISOR
026604	002407				BLT	;;CHECK THE SIGN
026606	003011				BGT	;;DIVISOR OF 0 IS A NO-NO
026610	052766	000003	000014		BIS	;;SET 'V' & 'C'
026616	012700	177777			MOV	;;SET REMAINDER TO ALL ONES
026622	000424				BR	;;EXIT
026624	005266	000002		2\$:	INC	;;KEEP TRACK OF DIVISORS SIGN
026630	000401				BR	
026632	005402			3\$:	NEG	;;NEGATE THE ORIGINAL NUMBER
026634	000241			4\$:	CLC	;;CLEAR 'C'
026636	000405				BR	;;START FORMING QUOTIENT
026640	006100			5\$:	ROL	;;POSITION MSB'S
026642	010003				MOV	;;COPY
026644	060203				ADD	;;COMPARE DIVIDEND & DIVISOR
026646	103001				BCC	;;BR IF DIVIDEND > DIVISOR
026650	010300				MOV	;;REMAINDER AFTER THIS LOOP

026652	006101		6\$:	ROL	R1	:::QUOTIENT BIT ENTERS HERE
026654	005316			DEC	(SP)	:::DONE?
026656	001370			BNE	5\$	:::BR IF NO
026660	005701			TST	R1	:::OVERFLOW?
026662	100005			BPL	8\$	:::BR IF NO
026664	052766	000002	000014	BIS	#2,14(SP)	:::SET 'V' IN RETURN STATUS WORD
026672	005000			CLR	R0	:::SFT REMAINDER TO ALL ZEROS
026674	010001		7\$:	MOV	R0,R1	:::COPY REMAINDER INTO QUOTIENT
026676	005726		8\$:	TST	(SP)+	:::CLEAR COUNTER FROM STACK
026700	005716			TST	(SP)	:::REMAINDER SIGN CORRECTION NEEDED?
026702	002004			BGE	9\$	:::BR IF NO
026704	005400			NEG	R0	:::NEGATE REMAINDER
026706	105066	000001		CLRB	1(SP)	:::CLEAR SIGN
026712	005316			DEC	(SP)	:::BUT DON'T FORGET QUOTIENT
026714	005726		9\$:	TST	(SP)+	:::QUOTIENT SIGN CORRECTION NEEDED?
026716	001401			BEQ	10\$	:::BR IF NO
026720	005401			NEG	R1	:::NEGATE QUOTIENT
026722	010166	000020	10\$:	MOV	R1,20(SP)	:::RETURN QUOTIENT AND
026726	010066	000016		MOV	R0,16(SP)	:::REMAINDER TO USER
026732	012603			MOV	(SP)+,R3	:::POP STACK INTO R3
026734	012602			MOV	(SP)+,R2	:::POP STACK INTO R2
026736	012601			MOV	(SP)+,R1	:::POP STACK INTO R1
026740	012600			MOV	(SP)+,R0	:::POP STACK INTO R0
026742	012666	000002		MOV	(SP)+,2(SP)	:::SETUP TO RETURN CONDITION CODES
026746	000002			RTI		:::RETURN

.SBTTL APT COMMUNICATIONS ROUTINE

```
*****
026750 112737 000001 027214 SATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
026756 112737 000001 027212 SATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
026764 000403 BR SATYC
026766 112737 000001 027214 SATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
026774 SATYC:
026774 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
026776 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
027000 105737 027212 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
027004 001450 BEQ 5$ ;;IF NOT: BR
027006 122737 000001 001254 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
027014 001031 BNE 3$ ;;IF NOT: BR
027016 132737 000100 001255 BITB #APTSPool,$ENVM ;;SHOULD SPOOL MESSAGES?
027024 001425 BEQ 3$ ;;IF NOT: BR
027026 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
027032 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
027040 005737 001234 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
027044 001375 BNE 1$ ;;IF NOT: WAIT
027046 010037 001250 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
027052 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
027054 001376 BNE 2$
027056 163700 001250 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
027062 006200 ASR R0 ;;GET MESSAGE LGTH IN WORDS
027064 010037 001252 MOV R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
027070 012737 000004 001234 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
027076 000413 BR 5$
027100 017637 000004 027124 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
027106 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
027114 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
027120 004737 023042 JSR PC,$TYPE ;;CALL TYPE MACRO
027124 000000 4$: .WORD 0
027126 5$:
027126 105737 027214 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
027132 001416 BEQ 12$ ;;IF NOT: BR
027134 005737 001254 TST $ENV ;;RUNNING UNDER APT?
027140 001413 BEQ 12$ ;;IF NOT: BR
027142 005737 001234 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
027146 001375 BNE 11$ ;;IF NOT: WAIT
027150 017637 000004 001236 MOV @4(SP),$FATAL ;;GET ERROR #
027156 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
027164 005237 001234 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
027170 105037 027214 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
027174 105037 027213 CLRB $LFLG ;;CLEAR LOG FLAG
027200 105037 027212 CLRB $MFLG ;;CLEAR MESSAGE FLAG
027204 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
027206 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
027210 000207 RTS PC ;;RETURN
027212 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
027213 000 $LFLG: .BYTE 0 ;;LOG FLAG
027214 000 $FFLG: .BYTE 0 ;;FATAL FLAG
               .EVEN
000200 APTSIZE = 200
000001 APTENV = 001
000100 APTSPool = 100
000040 APTCSUP = 040
```

```
2
3
4
5
6 027216 010046
7 027220 013700 040650
8 027224 016000 000010
9 027230 042700 177770
10 027234 136037 040636 001330
11 027242 001432
12 027244 105260 001472
13 027250 126037 001472 001316
14 027256 103424
15 027260 104401 001231
16 027264 104401 047625
17 027270 010046
    027272 104403
    027274 002
    027275 000
18 027276 104401 047634
19 027302 104401 047515
20 027306 104401 047645
21 027312 146037 040636 001330
22 027320 005337 021506
23 027324 000137 021334
24 027330 012600
25 027332 000207
26
27
28
29
30
31
32
33 027334 013704 040650
34 027340 012764 000040 000010
35 027346 042737 177770 001352
36 027354 013764 001352 000010
37 027362 000207
38
39
40
41
42
43
44
45
46 027364 005037 001326
47 027370 012737 027414 000004
48 027376 005037 000006
49 027402 005777 152126
50 027406 005237 001326
51 027412 000401
52 027414 022626
53 027416 012737 000006 000004
54 027424 000207

;THIS ROUTINE IS USED TO INCREMENT THE ERROR COUNT FOR EACH DRIVE BEING
;TESTED. IF THE TOTAL ERRORS ON ANY DRIVE EXCEEDS THE MAXIMUM ALLOWABLE
;ERRORS DESIGNATED IN LOCATION 'ERMAX', A MESSAGE WILL BE TYPED AND THE
;DRIVE IN ERROR WILL BE DROPPED FROM THE TEST.

INCEC: MOV R0, -(SP) ;SAVE R0
        MOV RMADR, R0 ;GET RM BASE ADDRESS
        MOV RMCS2(R0), R0 ;GET CONTENTS OF RMCS2
        BIC #C7, R0 ;SAVE UNIT SELECT BITS
        BITB ATABIT(R0), DRVSEL ;WAS THIS DRIVE SELECTED FOR TEST
        BEQ 1$ ;BR IF NO
        INCB ERRCN(R0) ;INCREMENT ERROR COUNT
        CMPB ERRCN(R0), ERMAX ;EXCEEDED ERROR LIMIT ?
        BLO 1$ ;BR IF NO
        TYPE , $CRLF ;CR-LF
        TYPE , MSDRIV ;TYPE 'DRIVE'
        MOV R0, -(SP) ;SAVE R0 FOR TYPEOUT
        TYPOS ;GO TYPE--OCTAL ASCII
        .BYTE 2 ;TYPE 2 DIGIT(S)
        .BYTE 0 ;SUPPRESS LEADING ZEROS
        TYPE , DROP ;TYPE 'DROPPED'
        TYPE , COMMA ;TYPE ','
        TYPE , EXCEED ;TYPE 'EXCEEDED MAXIMUM ERROR LIMIT'
        BICB ATABIT(R0), DRVSEL ;DESELECT DRIVE FROM TEST
        DEC $EOPCT ;ADJUST 'EOP' COUNT
        JMP $EOP ;RETURN TO $EOP
1$: MOV (SP)+, R0 ;RESTORE R0
    RTS PC

;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTER,
;AND DRIVERS, THEN SELECTS THE DRIVE.
;CALL:
;      JSR PC, CNTCLR ;CALL TO ROUTINE
;
;
CNTCLR: MOV RMADR, R4 ;GET RMCS1 BASE ADDRESS
        MOV #CLR, RMCS2(R4) ;ISSUE MASSBUS CLEAR AND
        BIC #C7, CHKDRV ;SAVE UNIT SELECT BITS
        MOV CHKDRV, RMCS2(R4) ;SELECT THE DRIVE
        RTS PC ;RETURN

;SET 'LPTAVL' TO THE PROPER STATE.
; LPTAVL = 0 IF NO LINE PRINTER AVAILABLE
; LPTAVL = 1 IF LINE PRINTER IS AVAILABLE
;CALL
;      JSR PC, LP.AVL
;
;      RETURN

LP.AVL: CLR LPTAVL ;START WITH NO PRINTER AVAILABLE
        MOV #1$, ERRVEC ;SETUP THE TIMEOUT VECTOR
        CLR ERRVEC+2
        TST @LPS ;IS THERE A LINE PRINTER?
        INC LPTAVL ;YES--SET AVAILABLE SWITCH
        BR 2$
1$: CMP (SP)+, (SP)+ ;NO--POP STACK
2$: MOV #ERRVEC+2, ERRVEC ;RESTORE TIMEOUT VECTOR
    RTS PC ;RETURN
```

```
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72 027426 010146
73 027430 012701 000006
74 027434 011146
75 027436 005011
76 027440 014146
77 027442 012711 027472
78 027446 005037 001342
79 027452 005777 152036
80 027456 012737 000001 001342
81 027464 004737 027574
82 027470 000414
83 027472 022626
84 027474 012711 027520
85 027500 005777 152022
86 027504 012737 177777 001342
87 027512 004737 027636
88 027516 000401
89 027520 022626
90 027522 012621
91 027524 012621
92 027526 012601
93 027530 032737 000100 001314
94 027536 001407
95 027540 012737 000020 001344
96 027546 012737 047040 001346
97 027554 000406
98 027556 012737 000016 001344
99 027564 012737 040432 001346
100 027572 000207
101
102 027574
103 027574 032737 000040 001314
104 027602 001014
105 027604 012777 027672 151676
106 027612 012777 000300 151672
107 027620 012777 000001 151670
108 027626 012777 000115 151660
109
110 027634 000207
111
```

```

:THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
:AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK
: 'CLKSTA' WILL INDICATE THE CLOCK TYPE
: 0= NO CLOCK
: +1= KW11-P
: -1= KW11-L
:THIS ROUTINE WILL ALSO SETUP 'TICKMS' (TIME
:PER CLOCK TICK IN MILLISECONDS) AND 'TICKUS'
:(TIME PER CLOCK TICK IN MICROSECONDS) AS
:PER SW00.
:SW00=0 -- 60HZ
:SW00=1 -- 50HZ
:CALL
:
:      JSR      PC,ST.CLK
:      RETURN
:
ST.CLK: MOV      R1,-(SP)          ;SAVE R1
        MOV      #ERRVEC+2,R1    ;SAVE AND SETUP TIMEOUT VECTOR
        MOV      (R1),-(SP)
        CLR      (R1)           ;LEVEL 0
        MOV      -(R1),-(SP)
        MOV      #1$, (R1)      ;GO TO 1$ ON TIMEOUT
        CLR      CLKSTA         ;SET CLOCK STATUS TO NO CLOCK
        TST      @PKCS          ;IS THERE A KW11-P?
        MOV      #1,CLKSTA      ;YES--SET STATUS TO KW11-P
        JSR      PC,ST.PCLK     ;START THE KW11-P
        BR       3$            ;GO TO EXIT
1$:      CMP      (SP)+,(SP)+    ;CLEAN UP THE STACK
        MOV      #2$, (R1)      ;IF TIMEOUT GO TO 2$
        TST      @LK$          ;IS THERE A KW11-L?
        MOV      #-1,CLKSTA     ;YES-- SET STATUS TO KW11-L
        JSR      PC,ST.LCLK     ;START THE KW11-L
        BR       3$            ;EXIT
2$:      CMP      (SP)+,(SP)+    ;CLEAN UP THE STACK
3$:      MOV      (SP)+,(R1)+    ;RESTORE THE TIMEOUT VECTOR
        MOV      (SP)+,(R1)+
        MOV      (SP)+,R1
        BIT      #SW06,C.SWR    ;RESTORE R1
        BEQ      4$            ;50HZ OR 60HZ?
        MOV      #20,TICKMS     ;BRANCH IF 60
        MOV      #20000.,TICKUS ;SETUP TIME PER
        BR       5$            ;TICK FOR 50HZ
4$:      MOV      #16,TICKMS     ;SETUP TIME PER
        MOV      #16666.,TICKUS ;TICK FOR 60HZ
5$:      RTS      PC            ;RETURN

ST.PCLK:
        BIT      #SW05,C.SWR    ;ALLOW SOFTWARE TIMEOUTS?
        BNE      1$            ;NO--BRANCH
        MOV      #SRVCLK,@PKV   ;SETUP THE KW11-P VECTOR
        MOV      #300,@PKV+2
        MOV      #1,@PKB        ;COUNT ONE TICK
        MOV      #115,@PKCS     ; 'INT.EN.', COUNT DOWN', 'MODE 1 (REPEAT)',
        ; 'LINE FREQ', AND 'RUN'
1$:      RTS      PC            ;RETURN
```

```

112 027636          ST.LCLK:
113 027636 032737 000040 001314 BIT    #SW05,C.SWR    ;ALLOW SOFTWARE TIMEOUTS?
114 027644 001011          BNE     1$              ;NO--BRANCH
115 027646 012777 027672 151646 MOV    #SRVCLK,@LKV    ;SETUP THE KW11-L VECTOR
116 027654 012777 000300 151642 MOV    #300,@LKV+2
117 027662 012777 000100 151636 MOV    #100,@LKS      ;START THE KW11-L
118 027670 000207          1$: RTS     PC            ;RETURN
119
120 027672 013746 001344          SRVCLK. MOV    TICKMS,-(SP) ;TIME PER TICK IN MILLISECONDS
121 027676 004737 044670          JSR     PC,RMTMR    ;COUNT THE ELAPSED TIME
122 027702 000002          RTI      ;RETURN AFTER INTERRUPT
123
124          ;THIS ROUTINE SETS UP DEFAULT PARAMETER VALUES WHEN THE PROGRAM IS
125          ;STARTED OR WHEN THE VALUE OF BIT00 IN 'C.SWR' IS CHANGED.
126          ;CALL
127          ; JSR     PC,LODFLT
128          ; RETURN
129
130 027704          LODFLT:
      C27704 010046          MOV    R0,-(SP)          ;;PUSH R0 ON STACK
      027706 010146          MOV    R1,-(SP)          ;;PUSH R1 ON STACK
      027710 010246          MOV    R2,-(SP)          ;;PUSH R2 ON STACK
      027712 010346          MOV    R3,-(SP)          ;;PUSH R3 ON STACK
131 027714 012737 166777 001332 MOV    #166777,TSTNMS ;SELECT TESTS 0-10,12,13 & 15-17
132 027722 012737 000003 001334 MOV    #3,TSTNMS+2   ;SELECT TESTS 20 & 21
133 027730 012700 002526          MOV    #DFLT,R0      ;DEFAULT PARAMETERS POINTER
134 027734 012701 003132          MOV    #PRMO,R1      ;TABLE POINTER
135 027740 010102          MOV    R1,R2              ;STOP ADDRESS
136 027742 012021          1$: MOV    (R0)+,(R1)+    ;MOVE DEFAULT PARAMETERS INTO
137 027744 020002          CMP     R0,R2              ;RUN TIME TABLES ** DONE?
138 027746 103775          BLO     1$                  ;NO--BRANCH
139 027750 012700 004176          MOV    #PAT8,R0      ;PATO DEFAULTS TO PATTERN 8
140 027754 012701 003576          MOV    #PAT0,R1
141 027760 012021          2$: MOV    (R0)+,(R1)+
142 027762 020027 004236          CMP    R0,#PAT9
143 027766 103774          BLO     2$
144 027770 032737 000001 001314 BIT    #BIT00,C.SWR   ;16 BIT MODE ?
145 027776 001012          BNE     3$                  ;BR IF 18 BIT MODE
146 030000 012737 000037 002470 MOV    #31,PRMLMT+24 ;SET 'FS' LIMIT TO 31.
147 030006 012737 000037 002472 MOV    #31,PRMLMT+26 ;SET 'LS' LIMIT TO 31.
148 030014 012737 160000 001452 MOV    #-<256.*32>,TRCKWC ;WORD COUNT FOR A 16 BIT TRACK
149 030022 000411          BR      4$                  ;CONTINUE
150 030024 012737 000035 002470 3$: MOV    #29,PRMLMT+24 ;SET 'FS' LIMIT TO 29.
151 030032 012737 000035 002472 MOV    #29,PRMLMT+26 ;SET 'LS' LIMIT TO 29.
152 030040 012737 161000 001452 MOV    #-<256.*30>,TRCKWC ;WORD COUNT FOR AN 18 BIT TRACK
153 030046 012701 002374          4$: MOV    #PRMPT,R1 ;ADDRESS OF PARAMETER POINTER TABLE
154 030052 005711          5$: TST     (R1)            ;END OF PARAMETER POINTER TABLE ?
155 030054 001425          BEQ     8$                  ;BR IF YES
156 030056 032731 004000          BIT    #BIT11,@(R1)+ ;IS 'LS' SELECTED AS A VARIABLE IN THIS TEST ?
157 030062 001773          BEQ     5$                  ;BR IF NO
158 030064 016102 177776          MOV    -2(R1),R2    ;GET FIRST POSITION IN PARAMETER TABLE
159 030070 011246          MOV    (R2),-(SP)          ;AND SAVE VARIABLES BITS THAT ARE USED.
160 030072 012703 000014          MOV    #12,R3       ;POSITION OF 'LS' IN TEST PARAMETER TABLE
161 030076 006216          6$: ASR     (SP)            ;IS THIS PARAMETER A VARIABLE ?
162 030100 103002          BCC     7$                  ;BR IF NO
163 030102 062702 000002          ADD    #2,R2        ;YES, POINT TO NEXT PARAMETER IN TABLE
164 030106 005303          7$: DEC     R3              ;AT 'LS' PARAMETER YET ?

```

```

165 030110 001372      BNE      6$      ;BR IF NO
166 030112 005726      TST      (SP)+    ;ADJUST THE STACK
167 030114 021237 002470 CMP      (R2),PRMLMT+24 ;IS 'LS' TOO LARGE FOR THE MODE SELECTED
168 030120 101754      BLOS     5$      ;BR IF NO
169 030122 013712 002470 MOV      PRMLMT+24,(R2) ;RESET VALUE FOR MODE USED
170 030126 000751      BR       5$      ;CONTINUE
171 030130      8$:      MOV      (SP)+,R3    ;POP STACK INTO R3
      030130 012603      MOV      (SP)+,R2    ;POP STACK INTO R2
      030132 012602      MOV      (SP)+,R1    ;POP STACK INTO R1
      030134 012601      MOV      (SP)+,R0    ;POP STACK INTO R0
      030136 012600      RTS      PC          ;RETURN
172 030140 000207
173
174      ;THIS ROUTINE FILLS THE PARAMETER TABLE THE CURRENT TEST.
175      ;CALL
176      :      MOV      #TESTNUM,$TSTNM ;LOAD THE TEST NUMBER
177      :      JSR      PC,LODPRM
178      :      RETURN
179
180 030142      LODPRM:
      030142 010146      MOV      R1,-(SP)    ;PUSH R1 ON STACK
      030144 010246      MOV      R2,-(SP)    ;PUSH R2 ON STACK
      030146 010346      MOV      R3,-(SP)    ;PUSH R3 ON STACK
      030150 010446      MOV      R4,-(SP)    ;PUSH R4 ON STACK
181 030152 005004      CLR      R4          ;CLEAR R4
182 030154 113704 001116 MOVB     $TSTNM,R4 ;GET THE TEST NUMBER
183 030160 006304      ASL      R4          ;SETUP TO ADDRESS WORDS
184 030162 016401 002374 MOV      PRMPT(R4),R1 ;GET THE TEST'S PARAMETER TABLE ADDRESS
185 030166 012702 002334 MOV      #PRM,R2    ;PARAMETER EXECUTION TABLE
186 030172 005003      CLR      R3          ;R3 IS USED AS A COUNTER
187 030174 013704 001352 MOV      CHKDRV,R4 ;GET DRIVE ADDRESS
188 030200 012122      MOV      (R1)+,(R2)+ ;LOAD PARAMETER SPECIFIER
189 030202 006237 002334 1$:      ASR      PRM    ;IS THIS PARAMETER USED IN THE TEST ?
190 030206 103002      BCC     2$      ;BR IF NOT
191 030210 012122      MOV      (R1)+,(R2)+ ;LOAD THE VALUE
192 030212 000401      BR       3$      ;CONTINUE
193 030214 005022      2$:      CLR      (R2)+ ;CLEAR THE UNUSED PARAMETER LOCATION
194 030216 005203      3$:      INC      R3    ;COUNT THE POSITION IN THE OUTPUT TABLE
195 030220 022702 002362 CMP      #PAT+2,R2 ;FINISHED ?
196 030224 001437      BEQ     7$      ;BR IF YES
197 030226 022703 000007 CMP      #7,R3    ;DOING TRACK PARAMETERS ?
198 030232 001363      BNE     1$      ;BR IF NO
199 030234 122764 000007 040542 CMPB   #7,DRVTP(R4) ;IS DEVICE AN RM05 ?
200 030242 001422      BEQ     6$      ;IF SO, OVERLAY FT, LT & IT WITH FT', LT' & IT'
201 030244 013737 002456 001374 MOV      PRMLMT+12,LSTRK ;GET LAST TRACK FOR AN RM03/2
202 030252 062703 000003 ADD      #3,R3    ;ADJUST COUNTER
203 030256 006237 002334 ASR      PRM    ;COUNT THE PARAMETER
204 030262 103001      BCC     4$      ;BR IF FT' IS NOT USED
205 030264 005721      TST     (R1)+    ;MOVE THE INPUT POINTER
206 030266 006237 002334 4$:      ASR      PRM    ;COUNT THE PARAMETER
207 030272 103001      BCC     5$      ;BR IF LT' NOT USED
208 030274 005721      TST     (R1)+    ;MOVE THE INPUT POINTER
209 030276 006237 002334 5$:      ASR      PRM    ;COUNT THE PARAMETER
210 030302 103337      BCC     1$      ;BR IF IT' NOT USED
211 030304 005721      TST     (R1)+    ;MOVE THE INPUT POINTER
212 030306 000735      BR       1$      ;KEEP GOING
213 030310 013737 002464 001374 6$:      MOV      PRMLMT+20,LSTRK ;GET LAST TRACK FOR AN RM05

```

```

214 030316 162702 000006          SUB    #6,R2          ;BACKUP THE OUTPUT POINTER
215 030322 000727          BR      1$          ;KEEP GOING
216 030324          7$:      MOV    (SP)+,R4          ;;POP STACK INTO R4
      030326 012604          MOV    (SP)+,R3          ;;POP STACK INTO R3
      030330 012602          MOV    (SP)+,R2          ;;POP STACK INTO R2
      030332 012601          MOV    (SP)+,R1          ;;POP STACK INTO R1
217 030334 000207          RTS     PC              ;RETURN
218
219          ;THIS ROUTINE LOADS A READ HEADER AND DATA COMMAND OR A SEEK COMMAND
220          ;INTO DPB.B+2 AND DPB.C+2, DEPENDING ON THE STATE OF 'CONTROL SWITCH'
221          ;BIT07.
222          ;CALL
223          ;      JSR     PC,LDCMD
224          ;      RETURN
225
226 030336 032737 000200 001314 LDCMD: BIT    #SW07,C.SWR      ;DO EXPLICIT SEEKS?
227 030344 001007          BNE     1$              ;YES--BRANCH
228 030346 012737 000173 047040          MOV    #READHD,DPB.B+2 ;NO--SET UP FOR READ HEADER AND
229 030354 012737 000173 047060          MOV    #RFADHD,DPB.C+2 ;DATA COMMAND
230 030362 000406          BR      2$
231 030364 012737 000105 047040 1$:      MOV    #SEEK,DPB.B+2    ;SETUP FOR SEEK COMMAND
232 030372 012737 000105 047060          MOV    #SEEK,DPB.C+2
233 030400 000207          2$:      RTS     PC
234
235          ;THIS ROUTINE WILL CALL THE RM05 DRIVER AND THEN WAIT ON THE FUNCTION
236          ;TO COMPLETE. IF AN ERROR OCCURS IT IS REPORTED.
237          ;CALL
238          ;      FILL 'DPB' WITH COMMAND INFORMATION
239          ;      JSR     R0,CALL.A
240          ;      RETURN
241
242 030402 005037 001222          CALL.A: CLR     $ESCAPE        ;NO ESCAPE ADDRESS
243 030406 004037 041406          JSR     R0,RM05          ;CALL RM05 DRIVER
244 030412 047016          DPB.A
245 030414 000772          BR      CALL.A
246 030416 005737 047034          1$:      TST     DPB.A+16      ;DONE?
247 030422 001775          BEQ     1$              ;NO--LOOP
248 030424 100050          BPL     5$              ;BRANCH IF NO ERROR
249 030426 012737 030522 001222          MOV    #3$,$ESCAPE      ;ESCAPE TO 3$ ON ERROR
250 030434 013737 047030 001366          MOV    DPB.A+12,CYL.DS    ;CYLINDER
      030442 113737 047027 001372          MOV    DPB.A+11,TRK.DS   ;TRACK
      030450 113737 047026 001370          MOV    DPB.A+10,SEC.DS   ;SECTOR
      030456 012746 047034          MOV    #DPB.A+16,-(SP)      ;STATUS/ERROR INDICATOR ADDRESS
      030462 004737 032170          JSR     PC,ERINDX          ;FORM DISPATCH INDEX
      030466 062607          ADD     (SP)+,PC              ;REPORT PROPER ERROR
      030470 104041          EMT     41                  ;NON-EXIST DRIVE
      030472 104042          EMT     42                  ;PARITY ERROR
      030474 104043          EMT     43                  ;UNSAFE ERROR
      030476 104044          EMT     44                  ;NON-I/O ERROR
251 030500 000240          NOP                      ;TO SYNC THE CALLING SEQ OF ERINDX
252 030502 005737 047154          TST     RM.REG+RMER1          ;ANY DRIVE ERROR ?
253 030506 001004          BNE     2$              ;BRANCH IF SO
254 030510 032737 100000 047202          BIT     #BSE,RM.REG+RMER2 ;BAD SPOT ERROR
255 030516 001013          BNE     5$              ;BRANCH IF SO
256 030520          2$:      EMT     45                  ;I/O ERROR
      030520 104045

```

```

257 030522 032737 040000 047202 3$: BIT #SKI, RM.REG+RMER2 ;SKI ERROR ?
258 030530 001402 4$: BEQ 4$ ;BR IF NO
259 030532 004037 032016 JSR R0, CALL.R ;DO RECALIBRATE COMMAND
260 030536 013746 047034 4$: MOV DPB.A+16, -(SP) ;STATUS WORD
261 030542 004737 032130 JSR PC, LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
262 030546 000200 5$: RTS R0 ;RETURN
263
264 ;THIS ROUTINE IS THE SAME AS 'CALL.A' EXCEPT FOR THE DPB USED AND IF
265 ;THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK,
266 ;AND SECTOR) READ IS CHECKED FOR VALIDITY.
267 ;CALL
268 ; FILL DPB
269 ; JSR R0, CALL.B
270 ; RETURN
271
272 030550 005037 001222 CALL.B: CLR $ESCAPE ;NO ESCAPE ADDRESS
273 030554 004037 041406 JSR R0, RM05 ;CALL RM05 DRIVER
274 030560 047036 DPB.B
275 030562 000772 BR CALL.B
276 030564 005737 047054 1$: TST DPB.B+16 ;DONE?
277 030570 001775 BEQ 1$ ;NO--BRANCH
278 030572 100055 BPL 5$ ;BRANCH IF NO ERROR
279 030574 012737 030700 001222 MOV #3$, $ESCAPE ;ESCAPE TO 3$ ON ERROR
280 030602 013737 047050 001366 MOV DPB.B+12, CYL.DS ;CYLINDER
030610 113737 047047 001372 MOV DPB.B+11, TRK.DS ;TRACK
030616 113737 047046 001370 MOV DPB.B+10, SEC.DS ;SECTOR
030624 012746 047054 MOV #DPB.B+16, -(SP) ;STATUS/ERROR INDICATOR ADDRESS
030630 004737 032170 JSR PC, ERINDX ;FORM DISPATCH INDEX
030634 062607 ADD (SP)+, PC ;REPORT PROPER ERROR
030636 104041 EMT 41 ;NON-EXIST DRIVE
030640 104042 EMT 42 ;PARITY ERROR
030642 104043 EMT 43 ;UNSAFE ERROR
030644 104044 EMT 44 ;NON-I/O ERROR
281 030646 000240 NOP ;TO SYNC THE CALLING SEQ OF ERINDX: RT.
282 030650 005737 047154 TST RM.REG+RMER1 ;DRIVE ERROR ?
283 030654 001404 BEQ 2$ ;BR IF NOT
284 030656 022737 000200 047154 CMP #HCE, RM.REG+RMER1 ;SEE IF ONLY 'HCE' SET
285 030664 001420 BEQ 5$ ;BR IF IT IS
286 030666 032737 100000 047202 2$: BIT #BSE, RM.REG+RMER2 ;BSE ERROR
287 030674 001033 BNE 7$ ;BRANCH IF SO
288 030676 104045 EMT 45 ;I/O ERROR
289 030700 032737 040000 047202 3$: BIT #SKI, RM.REG+RMER2 ;SKI ERROR ?
290 030706 001402 BEQ 4$ ;BR IF NO
291 030710 004037 032016 JSR R0, CALL.R ;DO RECALIBRATE COMMAND
292 030714 013746 047054 4$: MOV DPB.B+16, -(SP) ;STATUS WORD
293 030720 004737 032130 JSR PC, LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
294 030724 000410 BR 6$ ;CHECK FOR STALL
295 030726 123727 047040 000173 5$: CMPB DPB.B+2, #READHD ;DOING IMPLIED SEEKS?
296 030734 001004 BNE 6$ ;NO--BRANCH
297 030736 004037 032410 JSR R0, VERIFY ;YES--GO CHECK THE DATA
298 030742 047046 DPB.B+10
299 030744 000407 BR 7$ ;ERROR DURING VERIFY
300 030746 032737 040000 001314 6$: BIT #SW14, C.SWR ;STALL?
301 030754 001403 BEQ 7$ ;NO--BRANCH
302 030756 004037 032326 JSR R0, STALL ;YES--CALL STALL ROUTINE
303 030762 001454 .WORD STALL1 ;STALL TIME POINTER
304 030764 000200 7$: RTS R0 ;RETURN

```

```

305
306
307
308
309
310
311
312 030766 005037 001222
313 030772 004037 041406
314 030776 047056
315 031000 000772
316 031002 005737 047074
317 031006 001775
318 031010 100055
319 031012 012737 031116 001222
320 031020 013737 047070 001366
    031026 113737 047067 001372
    031034 113737 047066 001370
    031042 012746 047074
    031046 004737 032170
    031052 062607
    031054 104041
    031056 104042
    031060 104043
    031062 104044
321 031064 000240
322 031066 005737 047154
323 031072 001404
324 031074 022737 000200 047154
325 031102 001420
326 031104 032737 100000 047202
327 031112 001033
328 031114 104045
329 031116 032737 040000 047202
330 031124 001402
331 031126 004037 032016
332 031132 013746 047074
333 031136 004737 032130
334 031142 000410
335 031144 123727 047060 000173
336 031152 001004
337 031154 004037 032410
338 031160 047066
339 031162 000407
340 031164 032737 040000 001314
341 031172 001403
342 031174 004037 032326
343 031200 001454
344 031202 000200
345
346
347
348
349
350
351
352

```

:THIS ROUTINE IS THE SAME AS 'CALL.B' EXCEPT FOR THE DPB USED.
 :CALL
 : FILL DPB
 : JSR RO,CALL.C
 : RETURN
 :
 CALL.C: CLR \$ESCAPE ;NO ESCAPE ADDRESS
 JSR RO,RM05 ;CALL RM05 DRIVER
 DPB.C
 BR CALL.C
 1\$: TST DPB.C+16 ;DONE?
 BEQ 1\$;NO--LOOP
 BPL 5\$;YES--BRANCH IF NO ERROR
 MOV #3\$,\$ESCAPE ;ESCAPE TO 3\$ ON ERROR
 MOV DPB.C+12,CYL.DS ;CYLINDER
 MOVB DPB.C+11,TRK.DS ;TRACK
 MOVB DPB.C+10,SEC.DS ;SECTOR
 MOV #DPB.C+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
 JSR PC,ERINDX ;FORM DISPATCH INDEX
 ADD (SP)+,PC ;REPORT PROPER ERROR
 EMT 41 ;NON-EXIST DRIVE
 EMT 42 ;PARITY ERROR
 EMT 43 ;UNSAFE ERROR
 EMT 44 ;NON-I/O ERROR
 NOP ;TO SYNC THE CALLING SEQ OF ERINDX: RT.
 TST RM.REG+RMER1 ;DRIVE ERROR ?
 BEQ 2\$;BR IF NOT
 CMP #HCE,RM.REG+RMER1 ;SEE IF ONLY 'HCE' SET
 BEQ 5\$;BR IF YES
 2\$: BIT #BSE,RM.REG+RMER2 ;BSE ERROR ONLY ?
 BNE 7\$;BRANCH IF SO
 EMT 45 ;I/O ERROR
 3\$: BIT #SKI,RM.REG+RMER2 ;SKI ERROR ?
 BEQ 4\$;BR IF NO
 JSR RO,CALL.R ;DO RECALIBRATE COMMAND
 4\$: MOV DPB.C+16,-(SP) ;STATUS WORD
 JSR PC,LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
 BR 6\$
 5\$: CMPB DPB.C+2,#READHD ;DOING IMPLIED SEEK?
 BNE 6\$;NO--EXIT
 JSR RO,VERIFY ;YES--CHECK THE DATA
 DPB.C+10
 BR 7\$;ERROR DURING VERIFY
 6\$: BIT #SW14,C.SWR ;STALL?
 BEQ 7\$;NO--BRANCH
 JSR RO,STALL ;YES--CALL STALL ROUTINE
 .WORD STALL1 ;STALL TIME POINTER
 7\$: RTS RO

:THIS ROUTINE IS THE SAME AS 'CALL.A' EXCEPT FOR THE DPB USED AND
 :ON AN ERROR LOCATION 'ERR.CT' IS EXAMINED. IF ERR.CT IS EQUAL TO
 :SERFLG EXIT IS TO THE NEXT TEST.

:CALL
 : FILL DPB
 : JSR RO,DRVCAL

```

353      ; RETURN
354
355 031204 005037 001222      DRVCL: CLR $ESCAPE ;NO ESCAPE ADDRESS
356 031210 005037 001434      CLR WCEFLG ;CLEAR WRITE CHECK ERROR FLAG
357 031214 004037 041406      JSR RO,RM05 ;CALL RM05 DRIVER
358 031220 047076
359 031222 000770      DTADPB
BR      DRVCL
360
361 031224 005737 047114      DRVCL1: TST DTADPB+16 ;DONE
362 031230 001775      BEQ DRVCL1 ;NO--LOOP
363 031232 100402      BMI 1$ ;BR IF ERRORS
364 031234 000137 031776      JMP 15$ ;NO ERRORS
365 031240
1$:      MOV #3$, $ESCAPE ;;ESCAPE TO 3$ ON ERROR
366 031246 013737 047110 001366      MOV DTADPB+12,CYL.DS ;CYLINDER
031254 113737 047107 001372      MOV DTADPB+11,TRK.DS ;TRACK
031262 113737 047106 001370      MOV DTADPB+10,SEC.DS ;SECTOR
031270 012746 047114      MOV #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
031274 004737 032170      JSR PC,ERINDX ;FORM DISPATCH INDEX
031300 062607      ADD (SP)+,PC ;REPORT PROPER ERROR
031302 104041      EMT 41 ;NON-EXIST DRIVE
031304 104042      EMT 42 ;PARITY ERROR
031306 104043      EMT 43 ;UNSAFE ERROR
031310 104044      EMT 44 ;NON-I/O ERROR
367 031312 000240      NOP ;TO SYN THE CALLING SEQ OF THE ERINDX:
368 031314 005737 047154      TST RM.REG+RMER1 ;ANY DRIVE ERROR ?
369 031320 001011      BNE 2$ ;REPORT THE I/O ERROR IF SO
370 031322 032737 100000 047202      BIT #BSE,RM.REG+RMER2 ;BAD SPOT ERROR ?
371 031330 001405      BEQ 2$ ;BRANCH IF NOT
372 031332 012737 177777 001466      MOV #-1,BASFLG ;SET BAD SECTOR ENCOUNTER FLAG
373 031340 000137 031776      JMP 15$ ;OTHERWISE ,DON'T REPORT THE BSE
374 031344
2$:      EMT 45 ;I/O ERROR
375 031346 122737 000020 001116 3$:      CMPB #20,$TSTNM ;TEST 20?
376 031354 001170      BNE 11$ ;NO--BRANCH
377 031356 013746 047114      MOV DTADPB+16,-(SP) ;STATUS WORD
378 031362 004737 032130      JSR PC,LOP.CK ;SEE IF LOOP, ABCRT, OR CONTINUE
379 031366 122737 000151 047100      CMPB #WRCKD,DTADPB+2 ;DOING A WRITE CHECK?
380 031374 001172      BNE 13$ ;NO--BRANCH
381 031376 032737 040000 047150      BIT #BIT14,RM.REG+10 ;IS 'WCE'=1?
382 031404 001566      BEQ 13$ ;NO--BRANCH
383 031406 032777 000020 147540      BIT #SW04,@SWR ;INHIBIT WRITES?
384 031414 001162      BNE 13$ ;YES--BRANCH
385 031416 112737 000161 047100      MOVB #WRITE,DTADPB+2 ;SETUP FOR A WRITE
386 031424 005037 001222      CLR $ESCAPE ;NO ESCAPE ADDRESS
387 031430 004037 041406      JSR RO,RM05 ;DO THE WRITE
388 031434 047076      DTADPB
389 031436 000240      NOP
390 031440 005737 047114      4$:      TST DTADPB+16 ;DONE?
391 031444 001775      BEQ 4$ ;NO--LOOP
392 031446 100043      BPL 6$ ;YES--BRANCH IF NO ERROR
393 031450 012737 031736 001222      MOV #11$, $ESCAPE ;;ESCAPE TO 11$ ON ERROR
394 031456 013737 047110 001366      MOV DTADPB+12,CYL.DS ;CYLINDER
031464 113737 047107 001372      MOV DTADPB+11,TRK.DS ;TRACK
031472 113737 047106 001370      MOV DTADPB+10,SEC.DS ;SECTOR
031500 012746 047114      MOV #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
031504 004737 032170      JSR PC,ERINDX ;FORM DISPATCH INDEX

```

031510	062607			ADD	(SP)+,PC	:REPORT PROPER ERROR
031512	104041			EMT	41	:NON-EXIST DRIVE
031514	104042			EMT	42	:PARITY ERROR
031516	104043			EMT	43	:UNSAFE ERROR
031520	104044			EMT	44	:NON-I/O ERROR
395 031522	000240			NOP		:TO SYN THE CALLING SEQ OF ERINDX
396 031524	005737	047154		TST	RM.REG+RMER1	:ANY DRIVE ERROR ?
397 031530	001011			BNE	5\$	:BRANCH IF SO
398 031532	032737	100000	047202	BIT	#BSE, RM.REG+RMER2	:BAD SOPT ERROR
399 031540	001405			BEQ	5\$	:BRANCH IF NOT
400 031542	012737	177777	001466	MOV	#-1, BASFLG	:SET BAD SECTOR ENCOUNTER FLAG
401 031550	000137	031776		JMP	15\$	:EXIT
402 031554			5\$:			
031554	104045			EMT	45	:I/O ERROR
403 031556	112737	000151	047100	MOVB	#WRCKD, DTADPB+2	:COMMAND=WRITE CHECK DATA
404 031564	004037	041406		JSR	RO, RM05	:DO THE WRITE CHECK
405 031570	047076			DTADPB		
406 031572	000240			NOP		
407 031574	005737	047114	7\$:	TST	DTADPB+16	:DONE?
408 031600	001775			BEQ	7\$	:NO--LOOP
409 031602	100410			BMI	9\$	:YES--BRANCH IF ERROR
410 031604	004037	041406		JSR	RO, RM05	:DO A 2ND WRITE CHECK
411 031610	047076			DTADPB		
412 031612	000240			NOP		
413 031614	005737	047114	8\$:	TST	DTADPB+16	:DONE?
414 031620	001775			BEQ	8\$	:NO--LOOP
415 031622	100065			BPL	15\$	:YES--BRANCH IF NO ERROR
416 031624	012737	000001	001434	MOV	#1, WCEFLG	:SET THE WRITE CHECK ERROR FLAG
417 031632	012737	031736	001222	MOV	#11\$, \$ESCAPE	:ESCAPE TO 11\$ ON ERROR
418 031640	013737	047110	001366	MOV	DTADPB+12, CYL.DS	:CYLINDER
031646	113737	047107	001372	MOVB	DTADPB+11, TRK.DS	:TRACK
031654	113737	047106	001370	MOVB	DTADPB+10, SEC.DS	:SECTOR
031662	012746	047114		MOV	#DTADPB+16, -(SP)	:STATUS/ERROR INDICATOR ADDRESS
031666	004737	032170		JSR	PC, ERINDX	:FORM DISPATCH INDEX
031672	062607			ADD	(SP)+, PC	:REPORT PROPER ERROR
031674	104041			EMT	41	:NON-EXIST DRIVE
031676	104042			EMT	42	:PARITY ERROR
031700	104043			EMT	43	:UNSAFE ERROR
031702	104044			EMT	44	:NON-I/O ERROR
419 031704	000240			NOP		:SO SYN THE CALLING SEQ OF ERINDX:
420 031706	005737	047154		TST	RM.REG+RMER1	:ANY DRIVE ERROR
421 031712	001010			BNE	10\$	:BRANCH IF SO
422 031714	032737	100000	047202	BIT	#BSE, RM.REG+RMER2	:BAD SOPT ERROR ?
423 031722	001404			BEQ	10\$	:BRANCH IF NOT
424 031724	012737	177777	001466	MOV	#-1, BASFLG	:SET BAD SECTOR ENCOUNTER FLAG
425 031732	000421			BR	15\$	:OTHERWISE EXIT
426 031734			10\$:			
031734	104046			EMT	46	:REPORT THE FATAL WRITE CHECK ERROR
427 031736	032737	040000	047202	BIT	#SKI, RM.REG+RMER2	:SKI ERROR ?
428 031744	001402			BEQ	12\$	:BR IF NO
429 031746	004037	032016		JSR	RO, CALL.R	:DO RECALIBRATE COMMAND
430 031752	013746	047114	12\$:	MOV	DTADPB+16, -(SP)	:STATUS WORD
431 031756	004737	032130		JSR	PC, LOP.CK	:SEE IF LOOP, ABORT, OR CONTINUE
432 031762	123737	001464	001117	CMPIB	ERR.CT, \$ERFLG	:GO TO NEXT TEST?
433 031770	101002			BHI	15\$	:NO--BRANCH
434 031772	013700	001350		MOV	BYPASS, RO	:YES--GET EXIT ADDRESS
435 031776	032737	040000	001314	BIT	#SW14, C.SWR	:STALL?

```
436 032004 001403      BEQ      16$      ;NO--BRANCH
437 032006 004037 032326 JSR      R0,STALL ;YES--CALL STALL ROUTINE
438 032012 001456      .WORD    STALL2   ;STALL TIME POINTER
439 032014 000200      RTS      R0
440
441      ;THIS ROUTINE WILL ISSUE A RECALIBRATE COMMAND TO THE RM05 DRIVER
442      ;AND WAIT FOR THE FUNCTION TO COMPLETE.
443      ;CALL
444      ;
445      JSR      R0,CALL.R
446      RETURN
447 032016 005037 001222 CALL.R: CLR      $ESCAPE ;NO ESCAPE ADDRESS
448 032022 004037 041406 JSR      R0,RM05 ;CALL RM05 DRIVER
449 032026 047116      DPB.R
450 032030 000772      BR      CALL.R
451 032032 005737 047134 1$: TST      DPB.R+16 ;DONE?
452 032036 001775      BEQ      1$      ;NO--LOOP
453 032040 100032      BPL      3$      ;BRANCH IF NO ERROR
454 032042 012737 032116 001222 MOV      #2$, $ESCAPE ;ESCAPE TO 2$ ON ERROR
455 032050 013737 047130 001366 MOV      DPB.R+12,CYL.DS ;CYLINDER
      032056 113737 047127 001372 MOV      DPB.R+11,TRK.DS ;TRACK
      032064 113737 047126 001370 MOV      DPB.R+10,SEC.DS ;SECTOR
      032072 012746 047134      MOV      #DPB.R+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
      032076 004737 032170      JSR      PC,ERINDX ;FORM DISPATCH INDEX
      032102 062607      ADD      (SP)+,PC ;REPORT PROPER ERROR
      032104 104041      EMT      41 ;NON-EXIST DRIVE
      032106 104042      EMT      42 ;PARITY ERROR
      032110 104043      EMT      43 ;UNSAFE ERROR
      032112 104044      EMT      44 ;NON-I/O ERROR
456 032114 000240      NOP ;TO SYNC THE CALLING SEQ OF ERINDX
457 032116 013746 047034 2$: MOV      DPB.A+16,-(SP) ;STATUS WORD
458 032122 004737 032130      JSR      PC,LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
459 032126 000200      RTS      R0 ;RETURN
460
461      ;THIS SUBROUTINE CHECK FOR LOOP, ABORT, OR CONTINUE SWITCHES AFTER
462      ;ERRORS 41, 42, 43, 44, 45, AND 46.
463      ;CALL
464      ;
465      MOV      DTA+16,-(SP) ;STATUS WORD FROM DPB IN USE
466      JSR      PC,LOP.CK
467      RETURN
468 032130 032777 001000 147016 LOP.CK: BIT      #SW9,@SWR ;LOOP ON ERROR
469 032136 001402      BEQ      1$      ;BR IF NOT
470 032140 000177 146760      JMP      @SLPERR ;START AT THE LOOP ADDRESS
471 032144 005037 001222 1$: CLR      $ESCAPE ;CLEAR ERROR ESCAPE FLAG
472 032150 032766 072006 000002 BIT      #BIT14!BIT13!BIT12!BIT10!BIT02!BIT01,2(SP) ;CHECK ERROR TYPE
473 032156 001402      BEQ      2$      ;BR IF DRIVE NOT OFFLINE, UNLOADED, OR
      ;PERSISTENT UNSAFE OR FATAL MASSBUS PARITY
474
475 032160 000137 021334      JMP      $EOP ;TERMINATE DRIVE
476 032164 012616 2$: MOV      (SP)+,(SP) ;ADJUST RETURN ADDRESS
477 032166 000207      RTS      PC
478
479      ;THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
480      ;TO THE PROPER ERROR CALL. THE INDEX IS FORMED BY EXAMINING
481      ;THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB.
482      ;INDEX STATUS/ERROR
483      ;-----
```

```

484      :      0 BIT14!BIT13!BIT08!BIT01
485      :      2 BIT11!BIT10!BIT02
486      :      4 BIT12!BIT04
487      :      6 BIT05!BIT03.<BIT09 & COMMAND=NON-I/O>
488      :      10 BIT06.<BIT09 & COMMAND=I/O>
489      :CALL
490      :      JSR      #DPB+16,-(SP)      ;ADDRESS OF STATUS/ERROR INDICATOR
491      :      JSR      PC,ERINDX          ;FORM INDEX
492      :      RETURN                     ;INDEX IS ON THE STACK
493
494 032170 010046      ERINDX: MOV      R0,-(SP)      ;SAVE R0
495 032172 010146      MOV      R1,-(SP)      ;SAVE R1
496 032174 016600 000006      MOV      6(SP),R0      ;GET STATUS/ERROR INDICATOR POINTER
497 032200 011037 001356      MOV      (R0),SVSTAT    ;SAVE THE STATUS/ERROR INDICATOR
498 032204 005001      CLR      R1              ;START INDEX AT ZERO
499 032206 032710      BIT      (PC)+,(R0)        ;FORM INDEX OF 0?
500 032210 020402      .WORD    BIT13!BIT08!BIT01
501 032212 001037      BNE      5$              ;YES--BRANCH
502 032214 032710      BIT      (PC)+,(R0)        ;FORM PARITY ERROR OR PORT REQUEST INDEX (2)?
503 032216 006004      .WORD    BIT11!BIT10!BIT02
504 032220 001033      BNE      4$              ;YES--BRANCH
505 032222 032710      BIT      (PC)+,(R0)        ;FORM UNSAFE INDEX (4)?
506 032224 050020      .WORD    BIT14!BIT12!BIT04
507 032226 001027      BNE      3$              ;YES--BRANCH
508 032230 032710      BIT      (PC)+,(R0)        ;FORM NON-I/O ERROR INDEX (6)?
509 032232 000050      .WORD    BIT05!BIT03
510 032234 001023      BNE      2$              ;YES--BRANCH
511 032236 032710      BIT      (PC)+,(R0)        ;FORM I/O ERROR INDEX (10)?
512 032240 000100      .WORD    BIT06
513 032242 001017      BNE      1$              ;YES--BRANCH
514 032244 032710      BIT      (PC)+,(R0)        ;SOFTWARE TIMEOUT?
515 032246 001000      .WORD    BIT09
516 032250 001420      BEQ      5$              ;NO--FORM INDEX OF 0
517 032252 122760 000150 177762      CMPB   #150,-16(R0) ;YES--I/O?
518 032260 003011      BGT      2$              ;NO--BRANCH
519 032262 005737 047154      TST      RM.REG+RMER1    ;ANY DRIVE ERROR ?
520 032266 001005      BNE      1$              ;BRANCH IF SO
521 032270 032737 100000 047202      BIT      #BSE,RM.REG+RMER2 ;BSE ERROR
522 032276 001401      BEQ      1$              ;BRANCH IF NOT
523 032300 005201      INC      R1              ;SKIP , NOT REPORT BSE ERROR
524 032302 005201      1$: INC      R1              ;INDEX=10---ERROR=45 OR 46
525 032304 005201      2$: INC      R1              ;INDEX=6---ERROR=44
526 032306 005201      3$: INC      R1              ;INDEX=4---ERROR=43
527 032310 005201      4$: INC      R1              ;INDEX=2---ERROR=42
528 032312 006301      5$: ASL      R1              ;INDEX=0---ERROR=41
529 032314 010166 000006      MOV      R1,6(SP)      ;RETURN INDEX TO USER
530 032320 012601      MOV      (SP)+,R1          ;RESTORE R1
531 032322 012600      MOV      (SP)+,R0          ;RESTORE R0
532 032324 000207      RTS      PC              ;RETURN FROM CALL
533
534      :THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
535      :AMOUNT OF TIME IF BIT13 OF C.SWR = 0 OR A RANDOM AMOUNT OF TIME
536      :IF BIT 13 OF C.SWR = 1.
537      :STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0 - 7, AND STALL2
538      :CONTAINS THE TIME FOR TESTS 16-21.
539      :CALL
540      :      JSR      R0,STALL

```

```
541 ; TIME POINTER ; WHERE TO FIND THE STALL TIME
542
543 032326 013046 STALL: MOV @ (R0)+, -(SP) ; PICKUP STALL TIME
544 032330 032737 020000 001314 BIT #SW13, C.SWR ; USE A RANDOM TIME?
545 032336 001406 BEQ 1$ ; NO--BRANCH
546 032340 004737 026426 JSR PC, $RAND ; YES--FORM RANDOM NUMBER
547 032344 013716 026526 MOV $LONJM, (SP) ; AND USE IT FOR THE STALL TIME
548 032350 042716 177700 BIC #^C77, (SP) ; BUT NEVER > 64 MILLISECONDS
549 032354 005046 1$: CLR -(SP) ; CLEAR TEMP. LOCATION
550 032356 162766 000001 000002 2$: SUB #1, 2(SP) ; MORE STALL REQUIRED?
551 032364 103407 BLO 4$ ; NO--BRANCH
552 032366 012716 000144 MOV #100., (SP) ; STALL FOR ABOUT 1 MILLISECOND
553 032372 005700 3$: TST R0 ; NOP TO KILL TIME
554 032374 005366 000000 DEC 0(SP) ; COUNT
555 032400 001374 BNE 3$ ; LOOP IF MORE COUNTS NEEDED
556 032402 000765 BR 2$
557 032404 022626 4$: CMP (SP)+, (SP)+ ; CLEAN OFF THE STACK
558 032406 000200 RTS R0 ; EXIT
559
560 ; ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEKS
561 ; CALL
562 ; JSR R0, VERIFY
563 ; ADR POINTER ; ADDRESS OF DPB+10 (SECTOR NUMBER)
564 ; RETURN
565
566 032410 010146 VERIFY: MOV R1, -(SP) ; SAVE R1
567 032412 012001 MOV (R0)+, R1 ; GET ADDRESS OF DPB+10
568 032414 042737 150000 054522 BIC #150000, BUFFER ; STRIP FORMAT AND BAD SECTOR BITS FROM CYLINDER NUMBER
569 032422 023761 054522 000002 CMP BUFFER, 2(R1) ; CYLINDER NUMBER OK?
570 032430 001003 BNE 1$ ; NO--BRANCH
571 032432 023711 054524 CMP BUFFER+2, (R1) ; YES--HOW ABOUT TRACK/SECTOR?
572 032436 001441 BEQ 3$ ; BRANCH IF GOOD
573 032440 013737 054522 001360 1$: MOV BUFFER, CYL.RD ; SAVE THE EXPECTED AND THE
574 032446 113737 054525 001362 MOVB BUFFER+3, TRK.RD ; RECIEVED CYLINDER, TRACK,
575 032454 113737 054524 001364 MOVB BUFFER+2, SEC.RD ; AND SECTOR
576 032462 112137 001370 MOVB (R1)+, SEC.DS
577 032466 112137 001372 MOVB (R1)+, TRK.DS
578 032472 011137 001366 MOV (R1), CYL.DS
579 032476 012737 032510 001222 MOV #2$, $ESCAPE ; ; ESCAPE TO 2$ ON ERROR
580 032504 005740 TST -(R0) ; MAKE IT TEST PC+4
581 032506 104012 EMT 12 ; IMPROPER HEADER DATA
582 032510 012737 000107 047020 2$: MOV #RECAL, DPB.A+2 ; LOAD RECALIBRATE ORDER CODE
583 032516 004037 030402 JSR R0, CALL.A ; GO EXECUTE THE COMMAND
584 032522 005037 001222 CLR $ESCAPE ; CLEAR ERROR ESCAPE FLAG
585 032526 032777 001000 146420 BIT #SW9, @SWR ; LOOP ON ERROR ?
586 032534 001404 BEQ 4$ ; BR IF NOT
587 032536 000177 146362 JMP @SLPERR ; RETURN TO ERROR LOOP ADDRESS
588 032542 062700 000002 3$: ADD #2, R0 ; INCREMENT RETURN ADDRESS
589 032546 012601 4$: MOV (SP)+, R1 ; RESTORE R1
590 032550 000200 RTS R0 ; EXIT
591
592 ; THIS ROUTINE WILL PERFORM A 'MASSBUS' INIT. FOLLOWED BY
593 ; A 'RECALIBRATE' ON THE DRIVE UNDER TEST.
594 ; NOTE: THIS ROUTINE DESTROYS R1 AND R4
595 ; CALL
596 ; JSR R0, SRCH00 ; DO A MASSBUS INIT. AND RECAL
597 ; RETURN1 ; RETURN HERE IF NO ERROR
```

```

598          ;      RETURN2          ;RETURN HERE ON ERROR
599
600 032552 005001          SRCH00: CLR R1          ;INCASE OF ERROR (TYPTIM)
601 032554 005037 177776          CLR PS
602 032560 012777 043306 006064  MOV #ISR,@RMVEC          ;SETUP INTERRUPT VECTOR
603 032566 013704 040650          MOV RMADR,R4          ;PICKUP ADDRESS OF RMCS1
604 032572 012764 000040 000010  MOV #CLR,RMCS2(R4)          ;MASSBUS INIT.
605 032600 005037 047106          CLR DTADPB+10          ;TRACK=0; SECTOR=0
606 032604 005037 047110          CLR DTADPB+12          ;CYLINDER =0
607 032610 012737 000107 047100  MOV #RECAL,DTADPB+2          ;COMMAND = RECALIBRATE
608 032616 005037 001222          CLR $ESCAPE          ;NO ESCAPE ADDRESS
609 032622 004037 041406          JSR R0,RM05          ;CALL THE DRIVER
610 032626 047076          DTADPB          ;DPB POINTER
611 032630 000440          BR 4$          ;QUEUE IS FULL
612 032632 005737 047114          1$: TST DTADPB+16          ;WAIT ON DONE
613 032636 001775          BEQ 1$
614 032640 100030          BPL 3$          ;TAKE NORMAL EXIT IF NO ERROR
615 032642 012737 032716 001222  MOV #2$,$ESCAPE          ;:ESCAPE TO 2$ ON ERROR
616 032650 013737 047110 001366  MOV DTADPB+12,CYL.DS          ;CYLINDER
        032656 113737 047107 001372  MOV DTADPB+11,TRK.DS          ;TRACK
        032664 113737 047106 001370  MOV DTADPB+10,SEC.DS          ;SECTOR
        032672 012746 047114          MOV #DTADPB+16,-(SP)          ;STATUS/ERROR INDICATOR ADDRESS
        032676 004737 032170          JSR PC,ERINDX          ;FORM DISPATCH INDEX
        032702 062607          ADD (SP)+,PC          ;REPORT PROPER ERROR
        032704 104041          EMT 41          ;NON-EXIST DRIVE
        032706 104042          EMT 42          ;PARITY ERROR
        032710 104043          EMT 43          ;UNSAFE ERROR
        032712 104044          EMT 44          ;NON-I/O ERROR
        032714 104045          EMT 45          ;I/O ERROR
617 032716 005720          2$: TST (R0)+          ;ADJUST FOR ERROR EXIT
618 032720 000404          BR 4$          ;GO TO THE EXIT
619 032722 005064 000006          3$: CLR RMDA(R4)          ;TRACK AND SECTOR 0
620 032726 005064 000034          CLR RMDC(R4)          ;CYLINDER = 0
621 032732 000200          4$: RTS R0          ;RETURN
622
623          ;THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS & THE SERVO SETTLE DOWN TEST
624
625 032734 000002          DORTI: RTI          ;RETURN FROM INTERRUPT
626
627          ;THIS ROUTINE WILL INITIALIZE THE TIMERS USED BY THE *TIMING ROUTINES
628          ;CALL
629          ;      JSR PC,STRTMR
630          ;      RETURN
631
632 032736 104412          STRTMR: SAVREG          ;SAVE R0-R5
633 032740 012700 001376          MOV #TIM.UP,R0          ;START AT TIM.UP
634 032744 005020          1$: CLR (R0)+          ;CLEAR THE TIME TABLES
635 032746 020027 001432          CMP R0,#TIM.PT          ;DONE?
636 032752 103774          BLO 1$          ;NO--BRANCH
637 032754 012710 054522          MOV #BUFFER,(R0)          ;SETUP POINTER
638 032760 012737 077777 001376  MOV #^CBIT15,TIM.UP          ;SET MINIMUM TIME TO MAXIMUM
639 032766 012737 077777 001414  MOV #^CBIT15,TIM.DN          ;POSITIVE NUMBER
640 032774 104413          RESREG          ;RESTORE R0-R5
641 032776 000207          RTS PC          ;RETURN
642
643          ;THIS ROUTINE WILL ADD THE ELAPSED TIME TO THE AVERAGE COUNTER AND
644          ;MAINTAIN THE MINIMUM AND MAXIMUM TIMES.

```

```

645                                     ;NOTE: THIS ROUTINE DESTROYS R2
646                                     ;CALL
647                                     :      MOV      #TP,R3      ;PARAMETER POINTER
648                                     :      MOV      FLAG,R5      ;FLAG=0=COUNT UP
649                                     :                                     ;FLAG=-1=COUNT DOWN
650                                     :      JSR      PC,COUNT
651                                     :      RETURN
652
653 033000 012702 001376      COUNT:  MOV      #TIM.UP,R2      ;PICKUP THE 'UP' POINTER
654 033004 005705              TST      R5                  ;USE IT?
655 033006 001402              BEQ      1$                  ;YES--BRANCH
656 033010 012702 001414      MOV      #TIM.DN,R2      ;NO--PICKUP 'DOWN' POINTER
657 033014 027722 146500      1$:  CMP      @PKC,(R2)+      ;LESS THAN PREVIOUS LOW?
658 033020 002003              BGE      2$                  ;NO--BRANCH
659 033022 017762 146472 177776  MOV      @PKC,-2(R2)      ;YES--SAVE IT
660 033030 027763 146464 000004 2$:  CMP      @PKC,4(R3)      ;LESS THAN THE LOW LIMIT?
661 033036 002001              BGE      3$                  ;NO--BRANCH
662 033040 005212              INC      (R2)                ;YES--COUNT IT
663 033042 005722              TST      (R2)+              ;ADVANCE THE POINTER
664 033044 027722 146450      3$:  CMP      @PKC,(R2)+      ;GREATER THAN PREVIOUS HIGH?
665 033050 003403              BLE      4$                  ;NO--BRANCH
666 033052 017762 146442 177776  MOV      @PKC,-2(R2)      ;YES--SAVE IT
667 033060 027763 146434 000006 4$:  CMP      @PKC,6(R3)      ;GREATER THAN THE HIGH LIMIT?
668 033066 003401              BLE      5$                  ;NO--BRANCH
669 033070 005212              INC      (R2)                ;YES--COUNT IT
670 033072 005722              TST      (R2)+              ;ADVANCE THE POINTER
671 033074 067722 146420      5$:  ADD      @PKC,(R2)+      ;ADD THIS COUNT TO THE TOTAL
672 033100 005522              ADC      (R2)+
673 033102 005212              INC      (R2)                ;COUNT THIS READING
674 033104 022737 063052 001432      CMP      #BUFFER+<4*822.> ;TIM.PT ;SAVE THIS COUNT?
675 033112 101406              BLOS     6$                  ;NO--BRANCH
676 033114 017777 146400 146310  MOV      @PKC,@TIM.PT      ;YES--WELL SAVE IT THEN
677 033122 062737 000002 001432      ADD      #2,TIM.PT      ;ADVANCE THE POINTER
678 033130 000207      6$:  RTS      PC                  ;RETURN
679
680                                     ;THIS ROUTINE PRINTS THE SPEC OF ALL TIMING TESTS
681                                     ;CALL
682                                     :      JSR      R0,SPTYP
683                                     :      TABLE ADDRESS
684                                     :
685                                     :TABLE: .WORD    ASCII MESSAGE POINTER
686                                     :      .WORD    MIN VALUE
687                                     :      .WORD    MAX VALUE
688
689 033132 012002              SPTYP:  MOV      (R0)+,R2      ;THE TABLE ADDRESS
690 033134 032777 000100 146012  BIT      #SW06,@SWR      ;ALLOW PRINT
691 033142 001035              BNE      3$                  ;EXIT IF NOT
692 033144 104401 001231      TYPE     , $CRLF
693 033150 104401 001231      TYPE     , $CRLF
694 033154 012237 033162      MOV      (R2)+,1$
695 033160 104401              TYPE
696 033162 000000      1$:  .WORD    0
697 033164 012246      MOV      (R2)+,-(SP)      ;LOAD MIN VALUE
698 033166 001410      BEQ      2$                  ;SKIP IF MIN VALUE IS 0
699 033170 104401 050166      TYPE     ,MSGMIN
700 033174 004737 026136      JSR      PC,$SB2D      ;CONVERT TO DECIMAL
701 033200 004737 026366      JSR      PC,$SUPRS      ;TYPE IT

```

702 033204 104401 050210
703 033210 104401 050174
704 033214 011246
705 033216 004737 026136
706 033222 004737 026366
707 033226 104401 050210
708 033232 104401 001231
709 033236 000200

2\$: TYPE ,MSGOUS ;0 US
TYPE ,MSGMAX
MOV (R2),-(SP) ;MAXIMUM VALUE
JSR PC,\$\$SB2D
JSR PC,\$\$SUPRS
TYPE ,MSGOUS
TYPE ,\$CRLF ;CR-LF
3\$: RTS R0

;;THIS ROUTINE IS USED TO TYPE THE MINIMUM,
;;MAXIMUM, AND AVERAGE TIMES FOR THE TIMING TESTS
;;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 R0,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:

728 033240
033240 010246
033242 010346
033244 010446
033246 010546
729 033250 012002
730 033252 032777 000100 145674
731 033260 001154
732 033262 012237 033302
733 033266 012205
734 033270 012203
735 033272 011202
736 033274 012704 001376
737 033300 104401
738 033302 000000
739 033304 005764 000014
740 033310 001536
741 033312 104401 050166
742 033316 012446
033320 004737 026136
033324 004737 026366
743 033330 104401 050210
744 033334 005724
745 033336 001421
746 033340 104401 050345
747 033344 016446 177776
033350 004737 026136
033354 004737 026366
748 033360 104401 050215
749 033364 010346
033366 004737 026136

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 (R0)+,R2 ;PICKUP THE TABLE POINTER
BIT #SW06,@SWR ;INHIBIT TIME REPORTS?
BNE 9\$;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 8\$;NO--BRANCH
TYPE ,MSGMIN ;MIN=
MOV (R4)+,-(SP) ;PUT (R4)+ ON THE STACK
JSR PC,\$\$SB2D ;CHANGE TO DECIMAL ASCIZ
JSR PC,\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
TYPE ,MSGOUS ;"0 US"
TST (R4)+ ;ANY SEES BELOW THE LOW LIMIT
BEQ 3\$;NO--BRANCH
TYPE ,BLNKS2 ;TYPE 2 SPACES
MOV -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
JSR PC,\$\$SB2D ;CHANGE TO DECIMAL ASCIZ
JSR PC,\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
TYPE ,M BELOW ;"BELOW THE MINIMUM OF"
MOV R3,-(SP) ;PUT R3 ON THE STACK
JSR PC,\$\$SB2D ;CHANGE TO DECIMAL ASCIZ

```

033372 004737 026366 JSR PC,$SUPRS ;TYPE WITHOUT LEADING ZEROS
750 033376 104401 050210 TYPE ,MSGOUS
751 033402 104401 050174 3$: TYPE ,MSGMAX ;'MAX='
752 033406 012446 MOV (R4)+,-(SP) ;PUT (R4)+ ON THE STACK
033410 004737 026136 JSR PC,$SB2D ;CHANGE TO DECIMAL ASCII
033414 004737 026366 JSR PC,$SUPRS ;TYPE WITHOUT LEADING ZEROS
753 033420 104401 050210 TYPE ,MSGOUS
754 033424 005724 TST (R4)+ ;ANY SEES ABOVE THE HIGH LIMIT
755 033426 001421 BEQ 4$ ;NO--BRANCH
756 033430 104401 050345 TYPE ,BLNKS2 ;TYPE 2 SPACES
757 033434 016446 MOV -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
033440 004737 026136 JSR PC,$SB2D ;CHANGE TO DECIMAL ASCII
033444 004737 026366 JSR PC,$SUPRS ;TYPE WITHOUT LEADING ZEROS
758 033450 104401 050244 TYPE ,MABOVE ;'ABOVE THE MAXIMUM OF'
759 033454 010246 MOV R2,-(SP) ;PUT R2 ON THE STACK
033456 004737 026136 JSR PC,$SB2D ;CHANGE TO DECIMAL ASCII
033462 004737 026366 JSR PC,$SUPRS ;TYPE WITHOUT LEADING ZEROS
760 033466 104401 050210 TYPE ,MSGOUS
761 033472 104401 050202 4$: TYPE ,MSGAVG ;'AVG='
762 033476 012446 MOV (R4)+,-(SP) ;FORM THE AVERAGE
763 033500 012446 MOV (R4)+,-(SP)
764 033502 012446 MOV (R4)+,-(SP)
765 033504 004737 026530 JSR PC,$DIV
766 033510 006126 ROL (SP)+ ;IS THE REMAINDER OVER HALF?
767 033512 100001 BPL 5$ ;NO--BRANCH
768 033514 005216 INC (SP) ;YES--ROUND UP
769 033516 004737 026136 5$: JSR PC,$SB2D ;CHANGE TO DECIMAL ASCII
033522 004737 026366 JSR PC,$SUPRS ;TYPE WITHOUT LEADING ZEROS
770 033526 104401 050210 TYPE ,MSGOUS
771 033532 104401 050345 TYPE ,BLNKS2 ;TYPE 2 SPACES
772 033536 016446 MOV -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
033542 004737 026136 JSR PC,$SB2D ;CHANGE TO DECIMAL ASCII
033546 004737 026366 JSR PC,$SUPRS ;TYPE WITHOUT LEADING ZEROS
773 033552 022737 000012 001116 CMP #12,$TSTNM ;TEST 12?
774 033560 001403 BEQ 6$ ;BRANCH IF SO
775 033562 104401 050313 TYPE ,MSGNUM ;TYPE 'SEES TIMED'
776 033566 000402 BR 7$ ;TYPE IT
777 033570 104401 050273 6$: TYPE ,MSGSEA ;TYPE 'SEARCHES TIMED'
778
779 033574 010537 033302 7$: MOV R5,2$ ;NEXT MESSAGE POINTER
780 033600 001404 BEQ 9$ ;IF NONE EXIT
781 033602 005005 CLR R5 ;NO MORE THAN 2
782 033604 000635 BR 1$
783 033606 104401 050330 8$: TYPE ,MSGNON
784 033612 012605 9$: MOV (SP)+,R5 ;POP STACK INTO R5
033614 012604 MOV (SP)+,R4 ;POP STACK INTO R4
033616 012603 MOV (SP)+,R3 ;POP STACK INTO R3
033620 012602 MOV (SP)+,R2 ;POP STACK INTO R2
785 033622 000200 RTS R0 ;EXIT
786
787 ;THIS SUBROUTINE WILL INCREMENT THE TRACK
788 ;NUMBER (R2) BY THE AMOUNT SPECIFIED BY 'IT'.
789 ;CALL
790 JSR R0,INCRK
791 RETURN1 ;TRACK NUMBER GREATER THAN L*15

```

```

792      ;      RETURN2      ;TRACK NUMBER INCRFMENTED
793
794 033624 020237 002350      INCTRK: CMP      R2,LT      ;LAST TRACK COMPLETED?
795 033630 001410      BEQ      2$      ;YES--EXIT
796 033632 063702 002352      ADD      IT,R2      ;NO--UPDATE TRACK
797 033636 020237 002350      CMP      R2,LT      ;TRACK TO BIG?
798 033642 003402      BLE      1$      ;NO--EXIT
799 033644 013702 002350      MOV      LT,R2      ;YES--SET TRACK TO LAST TRACK
800 033650 005720      1$: TST      (R0)+      ;ADJUST FOR RETURN 2
801 033652 000200      2$: RTS      R0      ;RETURN
802
803
804      ;THIS SUBROUTINE WILL INCREMENT THE CYLINDER
805      ;NUMBER (R1) BY THE AMOUNT SPECIFIED BY 'IC'.
806      ;CALL
807      ;      JSR      R0,INCCYL
808      ;      RETURN1      ;CYLINDER NUMBER GREATER THAN LC15
809      ;      RETURN2      ;CYLINDER NUMBER INCREMENTED
810
811 033654 020137 002342      INCCYL: CMP      R1,LC      ;LAST CYLINDER COMPLETED?
812 033660 001410      BEQ      2$      ;YES--EXIT
813 033662 063701 002344      ADD      IC,R1      ;NO--UPDATE CYLINDER
814 033666 020137 002342      CMP      R1,LC      ;CYLINDER TO BIG?
815 033672 003402      BLE      1$      ;NO--EXIT
816 033674 013701 002342      MOV      LC,R1      ;YES--SET CYLINDER TO LAST CYLINDER
817 033700 005720      1$: TST      (R0)+      ;ADJUST FOR RETURN 2
818 033702 000200      2$: RTS      R0      ;RETURN
819
820      ;THIS ROUTINE DECREASES THE SECTOR ADDRESS.
821      ;CALL
822      ;      CLR      -(SP)      ;CLEAR THE STACK
823      ;      JSR      PC,DECSEC      ;SUBROUTINE ENTRY
824      ;      RETURN
825
826 033704 113766 047146 000002      DECSEC: MOVB     RM,REG+RMDA,2(SP)      ;PUT THE SECTOR ADDRESS ON THE STACK
827 033712 005366 000002      DEC      2(SP)      ;DECREMENT THE ADDRESS
828 033716 100003      BPL      1$      ;BR IF NOT CORRECTION NEEDED
829 033720 013766 002470 000002      MOV      PRMLMT+24,2(SP)      ;OVERFLOW OCCURED, FORCE TO MAXIMUM ADDRESS
830 033726 000207      1$: RTS      PC      ;RETURN
831
832      ;THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
833      ;WITH ADDRESSES FROM 0 TO 31 WITH EACH ADDRESS
834      ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
835      ;CALL
836      ;      JSR      PC,FILBUF
837      ;      RETURN
838
839 033730 104412      FILBUF: SAVREG      ;SAVE R0 - R5
840 033732 005000      CLR      R0      ;FIRST DISK ADDRESS
841 033734 012701 054522      MOV      #BUFFER,R1      ;START FILLING HERE
842 033740 012702 000400      1$: MOV      #256,R2      ;DO 256 WORDS
843 033744 010021      2$: MOV      R0,(R1)+      ;STORE
844 033746 005302      DEC      R2      ;MORE?
845 033750 003375      BGT      2$      ;YES--BRANCH
846 033752 005200      INC      R0      ;NO--UPDATE DISK ADDRESS
847 033754 023700 002470      CMP      PRMLMT+24,R0      ;DONE?
848 033760 103367      BHIS     1$      ;NO--BRANCH

```

```

849 033762 104413      RESREG      ;RESTORE R0 - R5
850 033764 000207      RTS          PC      ;RETURN
851
852      ;THIS ROUTINE WILL CLEAR THE BUFFER BY
853      ;SETTING EACH WORD TO '177400'.
854      ;CALL
855      ;      JSR      R0,CLRBUF
856      ;      RETURN
857
858 033766 104412      CLRBUF: SAVREG      ;SAVE R0 - R5
859 033770 012701 177400      MOV      #177400,R1      ;WORD TO FILL BUFFER WITH
860 033774 012702 054522      MOV      #BUFFER,R2      ;FIRST ADDRESS OF BUFFER
861 034000 012703 114522      MOV      #BUFFER+<512.*32>,R3      ;LAST ADDRESS+2 OF BUFFER
862 034004 010122      1$:      MOV      R1,(R2)+      ;FILL WORDS 1, 9,...249,...5625
863 034006 010122      MOV      R1,(R2)+      ;FILL WORDS 2,10,...250,...5626
864 034010 010122      MOV      R1,(R2)+      ;FILL WORDS 3,11,...251,...5627
865 034012 010122      MOV      R1,(R2)+      ;FILL WORDS 4,12,...252,...5628
866 034014 010122      MOV      R1,(R2)+      ;FILL WORDS 5,13,...253,...5629
867 034016 010122      MOV      R1,(R2)+      ;FILL WORDS 6,14,...254,...5630
868 034020 010122      MOV      R1,(R2)+      ;FILL WORDS 7,15,...255,...5631
869 034022 010122      MOV      R1,(R2)+      ;FILL WORDS 8,16,...256,...5632
870 034024 020203      CMP      R2,R3      ;DONE?
871 034026 103766      BLO      1$      ;NO--BRANCH
872 034030 104413      RESREG      ;RESTORE R0 - R5
873 034032 000200      RTS          R0      ;RETURN FROM CALL
874
875      ;THIS ROUTINE IS USED TO CHECK THE DATA BUFFER
876      ;FOR ADDRESSES 0 THROUGH 31 WITH EACH ADDRESS
877      ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
878      ;CALL
879      ;      JSR      R0,CKSCTR
880      ;      RETURN
881
882 034034 104412      CKSCTR: SAVREG      ;SAVE R0 - R5
883 034036 162706 000004      SUB      #4,SP      ;RESERVE TEMP. STORAGE AREA
884 034042 005001      CLR      R1      ;FIRST SECTOR
885 034044 012716 054522      MOV      #BUFFER,(SP)      ;FIRST ADDRESS OF DATA BUFFER
886 034050 005066 000002      CLR      2(SP)      ;NO ERRORS
887 034054 012702 000020      1$:      MOV      #16.,R2      ;LOOP COUNT (16*16=256)
888 034060 011603      MOV      (SP),R3      ;GET 1ST ADDRESS OF THIS SECTORS DATA
889 034062
893 034062 020123      2$:      CMP      R1,(R3)+      ;WORD 1
      BNE      7$      ;BRANCH IF BAD
      034064 001063      CMP      R1,(R3)+      ;WORD 2
      034066 020123      BNE      7$      ;BRANCH IF BAD
      034070 001061      CMP      R1,(R3)+      ;WORD 3
      034072 020123      BNE      7$      ;BRANCH IF BAD
      034074 001057      CMP      R1,(R3)+      ;WORD 4
      034076 020123      BNE      7$      ;BRANCH IF BAD
      034100 001055      CMP      R1,(R3)+      ;WORD 5
      034102 020123      BNE      7$      ;BRANCH IF BAD
      034104 001053      CMP      R1,(R3)+      ;WORD 6
      034106 020123      BNE      7$      ;BRANCH IF BAD
      034110 001051      CMP      R1,(R3)+      ;WORD 7
      034112 020123      BNE      7$      ;BRANCH IF BAD
      034114 001047      CMP      R1,(R3)+      ;WORD 8
      034116 020123      BNE      7$      ;BRANCH IF BAD
      034120 001045

```

034122	020123			CMP	R1,(R3)+	:WORD 9
034124	001043			BNE	7\$	:BRANCH IF BAD
034126	020123			CMP	R1,(R3)+	:WORD 10
034130	001041			BNE	7\$	:BRANCH IF BAD
034132	020123			CMP	R1,(R3)+	:WORD 11
034134	001037			BNE	7\$	:BRANCH IF BAD
034136	020123			CMP	R1,(R3)+	:WORD 12
034140	001035			BNE	7\$	:BRANCH IF BAD
034142	020123			CMP	R1,(R3)+	:WORD 13
034144	001033			BNE	7\$	:BRANCH IF BAD
034146	020123			CMP	R1,(R3)+	:WORD 14
034150	001031			BNE	7\$	:BRANCH IF BAD
034152	020123			CMP	R1,(R3)+	:WORD 15
034154	001027			BNE	7\$	:BRANCH IF BAD
034156	020123			CMP	R1,(R3)+	:WORD 16
034160	001025			BNE	7\$	:BRANCH IF BAD
894 034162	005302			DEC	R2	:FINISHED WITH THIS SECTORS DATA?
895 034164	001336			BNE	2\$	:NO--BRANCH
896 034166	062716	001000	3\$:	ADD	#512.,(SP)	:YES--FIRST ADDRESS OF NEXT SECTOR
897 034172	005201			INC	R1	:MOVE TO NEXT SECTOR
898 034174	023701	002470		CMP	PRMLMT+24,R1	:DONE?
899 034200	103325			BHIS	1\$	:NO--BRANCH
900 034202	005766	000002	4\$:	TST	2(SP)	:ERROR OCCUR?
901 034206	001406			BEQ	6\$	:NO--BRANCH
902 034210	123737	001464	00117	CMPB	ERR.CT,\$ERFLG	:MAX. ERROR OCCURRED?
903 034216	101002			BHI	6\$	:NO--BRANCH
904 034220	013700	001350	5\$:	MOV	BYPASS,R0	:TAKE ERROR EXIT
905 034224	062706	000004	6\$:	ADD	#4,SP	:FREE TEMP. AREA
906 034230	104413			RESREG		:RESTORE R0 - R5
907 034232	000200			RTS	R0	:RETURN FROM CALL
908 034234	010304		7\$:	MOV	R3,R4	:FORM WORD NUMBER AND
909 034236	161604			SUB	(SP),R4	:ADDRESS TO CONTINUE FROM
910 034240	010405			MOV	R4,R5	
911 034242	006204			ASR	R4	:WORD NUMBER
912 034244	042705	177740		BIC	#^C37,R5	
913 034250	001002			BNE	8\$	:BRANCH IF NOT A MULTIPLE OF 16
914 034252	012705	000040		MOV	#40,R5	:SET TO WORD 16
915 034256	006305		8\$:	ASL	R5	
916 034260	062705	034062		ADD	#2\$,R5	:ADDRESS
917 034264	016337	177776	001142	MOV	-2(R3),\$BDDAT	:SAVE BAD DATA
918 034272	005766	000002		TST	2(SP)	:FIRST ERROR?
919 034276	001015			BNE	10\$	:NO--BRANCH
920 034300	013737	047110	001366	MOV	DTADPB+12,CYL.DS	:CYLINDER NUMBER
921 034306	113737	047107	001372	MOV	DTADPB+11,TRK.DS	:TRACK NUMBER
922 034314	012737	034324	001222	MOV	#9\$,\$ESCAPE	:ESCAPE TO 9\$ ON ERROR
923 034322	104021			EMT	21	:DATA COMPARE FAILURE
924 034324	105166	000002	9\$:	COMB	2(SP)	:SET ERROR SWITCH
925 034330	000404			BR	11\$	
926						
927 034332			10\$:			
034332	012737	034342	001222	MOV	#11\$,\$ESCAPE	:ESCAPE TO 11\$ ON ERROR
928 034340	104022			EMT	22	:FOLLOWS EMT 21
929 034342	032777	001000	144604	BIT	#SW09,@SWR	:LOOP ON ERROR?
930 034350	001323			BNE	5\$	:YES
931 034352	032777	000002	144574	BIT	#SW01,@SWR	:STOP DATA COMPARE?
932 034360	001310			BNE	4\$	:YES--BRANCH
933 034362	123737	001464	001117	CMPB	ERR.CT,\$ERFLG	:MAX. ERRORS?

```

934 034370 101713      BLOS 5$      ;YES--BRANCH
935 034372 032777 000040 144554  BII  #SW05,@SWR ;REPORT ONLY 1ST ERROR PER SECTOR?
936 034400 001272      BNE 3$      ;YES--BRANCH
937 034402 000115      JMP (R5)
938
939      ;THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
940      ;DESIRED PATTERN INTO THE DATA BUFFER.
941      ;CALL
942      :      MOV #NX,R0      ;PATTERN NUMBER INDEX TO R0
943      :      JSR PC,SETBUF
944
945 034404 104412      SETBUF: SAVREG      ;SAVE R0 - R5
946 034406 012701 054522      MOV #BUFFER,R1      ;FIRST ADDRESS
947 034412 013702 047102      MOV DTADPB+4,R2      ;WORD COUNT
948 034416 016003 003536      1$: MOV PAT.PT(R0),R3      ;PICKUP PATTERN POINTER
949 034422 012321      MOV (R3)+,(R1)+      ;MOVE WORD 1 INTO DATA BUFFER
950 034424 012321      MOV (R3)+,(R1)+      ;MOVE WORD 2 INTO DATA BUFFER
951 034426 012321      MOV (R3)+,(R1)+      ;MOVE WORD 3 INTO DATA BUFFER
952 034430 012321      MOV (R3)+,(R1)+      ;MOVE WORD 4 INTO DATA BUFFER
953 034432 012321      MOV (R3)+,(R1)+      ;MOVE WORD 5 INTO DATA BUFFER
954 034434 012321      MOV (R3)+,(R1)+      ;MOVE WORD 6 INTO DATA BUFFER
955 034436 012321      MOV (R3)+,(R1)+      ;MOVE WORD 7 INTO DATA BUFFER
956 034440 012321      MOV (R3)+,(R1)+      ;MOVE WORD 8 INTO DATA BUFFER
957 034442 012321      MOV (R3)+,(R1)+      ;MOVE WORD 9 INTO DATA BUFFER
958 034444 012321      MOV (R3)+,(R1)+      ;MOVE WORD 10 INTO DATA BUFFER
959 034446 012321      MOV (R3)+,(R1)+      ;MOVE WORD 11 INTO DATA BUFFER
960 034450 012321      MOV (R3)+,(R1)+      ;MOVE WORD 12 INTO DATA BUFFER
961 034452 012321      MOV (R3)+,(R1)+      ;MOVE WORD 13 INTO DATA BUFFER
962 034454 012321      MOV (R3)+,(R1)+      ;MOVE WORD 14 INTO DATA BUFFER
963 034456 012321      MOV (R3)+,(R1)+      ;MOVE WORD 15 INTO DATA BUFFER
964 034460 012321      MOV (R3)+,(R1)+      ;MOVE WORD 16 INTO DATA BUFFER
965 034462 062702 000020      ADD #16,R2 ;DONE?
966 034466 001353      BNE 1$      ;NO--BRANCH
967 034470 104413      RESREG
968 034472 000207      RTS PC      ;RESTORE R0 - R5
969      ;RETURN
970
971      ;THIS ROUTINE COMPARES A 16 WORD DATA PATTERN
972      ;AGAINST THE DATA BUFFER
973      ;CALL
974      :      MOV #NX,R0      ;PATTERN NUMBER INDEX TO R0
975      :      JSR PC,DATCMP
976      :      RETURN
977
978 034474 104412      DATCMP: SAVREG      ;SAVE R0 - R5
979 034476 012701 054522      MOV #BUFFER,R1      ;FIRST ADDRESS OF BUFFER
980 034502 013702 047102      MOV DTADPB+4,R2      ;WORD COUNT
981 034506 005046      CLR -(SP)      ;NO ERROR
982 034510 016003 003536      1$: MOV PAT.PT(R0),R3      ;PATTERN POINTER
983 034514      2$:
984 034516 001044      SUB (R3)+,(R1)+      ;CHECK WORD 1
985 034520 162321      BNE 4$      ;BRANCH IF DIFFERENT
986 034522 001042      SUB (R3)+,(R1)+      ;CHECK WORD 2
987 034524 162321      BNE 4$      ;BRANCH IF DIFFERENT
988 034526 001040      SUB (R3)+,(R1)+      ;CHECK WORD 3
989 034530 162321      BNE 4$      ;BRANCH IF DIFFERENT
990 034532 001036      SUB (R3)+,(R1)+      ;CHECK WORD 4
991      BNE 4$      ;BRANCH IF DIFFERENT

```

034534	162321		SUB	(R3)+,(R1)+	:CHECK WORD 5
034536	001034		BNE	4\$	:BRANCH IF DIFFERENT
034540	162321		SUB	(R3)+,(R1)+	:CHECK WORD 6
034542	001032		BNE	4\$	:BRANCH IF DIFFERENT
034544	162321		SUB	(R3)+,(R1)+	:CHECK WORD 7
034546	001030		BNE	4\$	:BRANCH IF DIFFERENT
034550	162321		SUB	(R3)+,(R1)+	:CHECK WORD 8
034552	001026		BNE	4\$	:BRANCH IF DIFFERENT
034554	162321		SUB	(R3)+,(R1)+	:CHECK WORD 9
034556	001024		BNE	4\$	:BRANCH IF DIFFERENT
034560	162321		SUB	(R3)+,(R1)+	:CHECK WORD 10
034562	001022		BNE	4\$	:BRANCH IF DIFFERENT
034564	162321		SUB	(R3)+,(R1)+	:CHECK WORD 11
034566	001020		BNE	4\$	:BRANCH IF DIFFERENT
034570	162321		SUB	(R3)+,(R1)+	:CHECK WORD 12
034572	001016		BNE	4\$	:BRANCH IF DIFFERENT
034574	162321		SUB	(R3)+,(R1)+	:CHECK WORD 13
034576	001014		BNE	4\$	:BRANCH IF DIFFERENT
034600	162321		SUB	(R3)+,(R1)+	:CHECK WORD 14
034602	001012		BNE	4\$	:BRANCH IF DIFFERENT
034604	162321		SUB	(R3)+,(R1)+	:CHECK WORD 15
034606	001010		BNE	4\$	:BRANCH IF DIFFERENT
034610	162321		SUB	(R3)+,(R1)+	:CHECK WORD 16
034612	001006		BNE	4\$	:BRANCH IF DIFFERENT
973 034614	062702	000020	ADD	#16.,R2	:DONE ?
974 034620	001333		BNE	1\$	:NO--BRANCH
975 034622	005726		TST	(SP)+	:YES -- CLEAN UP STACK
976 034624	104413		RESREG		:RESTORE R0 - R5
977 034626	000207		RTS	PC	
978					
979 034630	010104		MOV	R1,R4	:FORM THE WORD NUMBER
980 034632	162704	054522	SUB	#BUFFER,R4	
981 034636	006204		ASR	R4	:WORD NUMBER
982 034640	010305		MOV	R3,R5	:FORM ADDRESS TO CONTINUE FROM
983 034642	166005	003536	SUB	PAT.PI(R0),R5	
984 034646	006305		ASL	R5	
985 034650	062705	034514	ADD	#2\$,R5	:ADDRESS
986 034654	064341		ADD	-(R3),-(R1)	:RECONSTRUCT THE BAD WORD
987 034656	010137	001136	MOV	R1,\$BDADR	:SAVE THE ERROR INFORMATION
988 034662	010337	001134	MOV	R3,\$GDADR	
989 034666	012137	001142	MOV	(R1)+,\$BDDAT	
990 034672	012337	001140	MOV	(R3)+,\$GDDAT	
991 034676	005716		TST	(SP)	:1ST DATA COMPARE ERROR?
992 034700	001023		BNE	6\$	:NO--BRANCH
993 034702	013737	047110 001366	MOV	DTADPB+12,CYL.DS	:CYLINDER
994 034710	113737	047107 001372	MOVB	DTADPB+11,TRK.DS	:TRACK
995 034716	113737	047106 001370	MOVB	DTADPB+10,SEC.DS	:SECTOR
996 034724	016600	000026	MOV	26(SP),R0	:GET TEST PC+4
997 034730	012737	034740 001222	MOV	#5\$,\$ESCAPE	:ESCAPE TO 5\$ ON ERROR
998 034736	104013		EMT	13	:DATA COMPARE FAILURE
999 034740	016600	000020	MOV	20(SP),R0	:PATTERN NUMBER INDEX
1000 034744	105116		COMB	(SP)	:SET THE ERROR SWITCH
1001 034746	000404		BR	7\$	
1002					
1003 034750					
1004 034750	012737	034760 001222	MOV	#7\$,\$ESCAPE	:ESCAPE TO 7\$ ON ERROR
1004 034756	104014		EMT	14	:FOLLOWS EMT 13

```

1005 034760 032777 000002 144166 7$: BIT #JW01,@SWR ;STOP DATA COMPARE?
1006 034766 001315 BNE 3$ ;YES--EXIT
1007 034770 123737 001464 001117 CMPB ERR.CT,$ERFLG ;MAX. ERRORS?
1008 034776 101004 BHI 8$ ;NO--BRANCH
1009 035000 013766 001350 000002 MOV BYPASS,2(SP) ;YES--ERROR EXIT
1010 035006 000705 BR 3$
1011 035010 000115 8$: JMP (R5) ;NO--CONTINUE AT NEXT WORD
1012
1013 ;THIS ROUTINE WILL FILL THE DATA BUFFER (256*32 WORDS) WITH
1014 ;A RANDOM PATTERN. THE FIRST TWO WORDS OF EVERY 256 WILL
1015 ;BE THE BASE OF THE RANDOM NUMBER GENERATOR FOR THE
1016 ;NEXT 254 WORDS.
1017 ;NOTE: THIS ROUTINE DESTROYS R1 AND R2
1018 ;CALL
1019 ; JSR R0,FILRAN
1020 ; RETURN
1021
1022 035012 012701 054522 FILRAN: MOV #BUFFER,R1
1023 035016 013702 002470 MOV PRMLMT+24,R2 ;MAXIMUM NUMBER OF SECTORS
1024 035022 004037 035232 1$: JSR R0,RANPAT
1025 035026 005302 DEC R2
1026 035030 100374 BPL 1$
1027 035032 000200 RTS R0
1028
1029 ;THIS ROUTINE USES THE FIRST TWO WORDS OF THE
1030 ;READ BUFFER TO GENERATED A RANDOM PATTERN. THEN
1031 ;THE READ BUFFER IS COMPARED TO THE PATTERN GENERATED.
1032 ;NOTE: THIS ROUTINE DESTROYS R1-R4
1033 ;CALL
1034 ; JSR R0,RANCK
1035 ; RETURN
1036
1037 035034 013746 026524 RANCK: MOV $HINUM,-(SP) ;SAVE THE PRESENT RANDOM NUMBER
1038 035040 013746 026526 MOV $LONUM,-(SP)
1039 035044 012702 055522 MOV #BUFFER+512.,R2 ;READ BUFFER ADDRESS
1040 035050 012701 056522 MOV #BUFFER+1024.,R1 ;RANDOM PATTERN ADDRESS
1041 035054 010103 MOV R1,R3 ;COPY IT INTO R3 FOR LATER USE
1042 035056 011237 026526 MOV (R2),$LONUM ;PRIME THE RANDOM NUMBER GENERATOR
1043 035062 016237 000002 026524 MOV 2(R2),$HINUM
1044 035070 004037 035232 JSR R0,RANPAT ;GENERATE A RANDOM PATTERN
1045 035074 012637 026526 MOV (SP)+,$LONUM ;RESTORE PRESENT RANDOM NUMBER
1046 035100 012637 026524 MOV (SP)+,$HINUM
1047 035104 005046 CLR -(SP) ;NO ERRORS
1048 035106 162322 1$: SUB (R3)+,(R2)+ ;ARE THESE TWO WORDS DIFFERENT?
1049 035110 001441 BEQ 4$ ;NO--BRANCH
1050 035112 012737 035164 001222 MOV #3,$$ESCAPE ;ESCAPE TO 3$ ON ERROR
1051 035120 064342 ADD -(R3),-(R2) ;RECREATE THE BAD WORD
1052 035122 010237 001136 MOV R2,$BDADR ;ADDRESS OF BAD DATA
1053 035126 010337 001134 MOV R3,$GDADR ;ADDRESS OF GOOD DATA
1054 035132 012237 001142 MOV (R2)+,$BDDAT ;BAD DATA
1055 035136 012337 001140 MOV (R3)+,$GDDAT ;GOOD DATA
1056 035142 010204 MOV R2,R4 ;FORM WORD NUMBER (1 TO 256)
1057 035144 162704 055522 SUB #BUFFER+512.,R4
1058 035150 006204 ASR R4
1059 035152 005716 TST (SP) ;FIRST ERROR
1060 035154 001002 BNE 2$ ;NO--BRANCH
1061 035156 105116 COMB (SP) ;YES--SET ERROR SWITCH

```

```

1062 035160 104015      EMT      15      ;DATA COMPARE FAILURE
1063 035162      2$:      EMT      16      ;FOLLOWS EMT 15
      035162 104016      BIT      #SW09,@SWR ;LOOP ON ERROR?
1064 035164 032777 001000 143762 3$:      BNE      5$      ;YES--BRANCH
1065 035172 001012      CMPB     ERR.CT,$ERFLG ;MAX. ERRORS OCCURRED?
1066 035174 123737 001464 001117      BLOS     5$      ;YES--BRANCH
1067 035202 101406      BIT      #SW01,@SWR ;STOP COMPARING?
1068 035204 032777 000002 143742      BNE      5$      ;YES--BRANCH
1069 035212 001002      4$:      LMP      R1,R3      ;ALL DATA BEEN COMPARED?
1070 035214 020103      BHI      1$      ;NO--BRANCH
1071 035216 101333      5$:      TST      (SP)+      ;ERROR OCCUR?
1072 035220 005726      BEQ      6$      ;NO--BRANCH
1073 035222 001402      MOV      BYPASS,R0      ;TAKE ERROR EXIT
1074 035224 013700 001350      6$:      RTS      R0      ;EXIT
1075 035230 000200
1076
1077      ;THIS ROUTINE FILLS A 256 WORD BUFFER WITH A RANDOM
1078      ;PATTERN OF WHICH THE FIRST TWO WORDS ARE THE BASE
1079      ;OF THE PATTERN.
1080      ;CALL
1081      ;      MOV      #ADR,R1      ;ADDRESS OF THE BUFFER
1082      ;      JSR      R0,RANPAT
1083      ;      RETURN
1084
1085 035232 010246      RANPAT: MOV      R2,-(SP)      ;SAVE R2
1086 035234 012702 000200      MOV      #256/2.,R2      ;GENERATE 256 WORDS
1087 035240 000402      BR      2$
1088 035242 004737 026426 1$:      JSR      PC,$RAND      ;GENERATE A RANDOM NUMBER
1089 035246 013721 026526 2$:      MOV      $LONUM,(R1)+      ;PUT LOW WORD IN BUFFER
1090 035252 013721 026524      MOV      $HINUM,(R1)+      ;PUT HIGH WORD IN BUFFER
1091 035256 005302      DEC      R2      ;DONE?
1092 035260 003370      BGT      1$      ;NO--BRANCH
1093 035262 012602      MOV      (SP)+,R2      ;RESTORE R2
1094 035264 000200      RTS      R0      ;EXIT
1095
1096      ;THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR
1097      ;ADDRESSES AND SAVES THEN IN THE DPB (DTADPB+10, 11 & DTADPB+12).
1098      ;NOTE: THIS ROUTINE DESTROYS R1-R3
1099      ;CALL
1100      ;      JSR      R0,RANADR
1101      ;      RETURN
1102
1103 035266 004737 026426      RANADR: JSR      PC,$RAND      ;GENERATE A RANDOM NUMBER
1104 035272 113701 026526      MOVB     $LONUM,R1      ;FORM SECTOR IN R1
1105 035276 113702 026527      MOVB     $LONUM+1,R2      ;FORM TRACK IN R2
1106 035302 013703 026524      MOV      $HINUM,R3      ;FORM CYLINDER IN R3
1107 035306 105701      TSTB     R1      ;ENSURE THE SECTOR IS BETWEEN 0 AND 31
1108 035310 002403      BLT      2$
1109 035312 123701 002470 1$:      CMPB     PRMLMT+24,R1      ;CHECK MAXIMUM SECTOR ADDRESS
1110 035316 103003      BHIS     3$
1111 035320 000241      2$:      CLC
1112 035322 106001      RORB     R1
1113 035324 000772      BR      1$
1114 035326 105702      3$:      TSTB     R2      ;ENSURE THE TRACK IS BETWEEN 0 AND LAST TRACK
1115 035330 002403      BLT      5$
1116 035332 123702 001374 4$:      CMPB     LSTRK,R2
1117 035336 002003      BGE      6$

```

```

1118 035340 000241      5$:    CLC
1119 035342 106002      RORB    R2
1120 035344 000772      BR      4$
1121 035346 023703 002340  6$:    CMP    FC,R3      ;ENSURE THE CYLINDER IS BETWEEN FC AND LC
1122 035352 003413      BLE    7$
1123 035354 000241      CLC
1124 035356 006003      ROR     R3
1125 035360 005503      ADC     R3
1126 035362 001371      BNE     6$
1127 035364 010103      MOV     R1,R3
1128 035366 000303      SWAB   R3
1129 035370 060203      ADD     R2,R3
1130 035372 005203      INC     R3
1131 035374 003364      BGT     6$
1132 035376 005403      NEG     R3
1133 035400 000762      BR      6$
1134 035402 023703 002342  7$:    CMP    LC,R3
1135 035406 002003      BGE     8$
1136 035410 000241      CLC
1137 035412 006003      ROR     R3
1138 035414 000772      BR      7$
1139 035416 023703 002340  8$:    CMP    FC,R3
1140 035422 003403      BLE    9$
1141 035424 005203      INC     R3
1142 035426 000303      SWAB   R3
1143 035430 000764      BR      7$
1144 035432 110137 047106  9$:    MOVB   R1,DTADPB+10 ;SAVE SECTOR ADDRESS
1145 035436 110237 047107      MOVB   R2,DTADPB+11 ;SAVE TRACK ADDRESS
1146 035442 010337 047110      MOV     R3,DTADPB+12 ;SAVE CYLINDER ADDRESS
1147 035446 000200      RTS     R0      ;RETURN
1148
1149      ;THIS ROUTINE IS USED TO INPUT THE "CONTROL SWITCHES".
1150      ;IF SWR<07>=1 THE PRESENT SETTING WILL BE TYPED AND THE NEW
1151      ;SETTING IS READ AND STORED.
1152      ;NOTE: THIS ROUTINE DESTROYS R3 AND R4
1153      ;CALL
1154      ;      JSR     PC,GETSWR
1155      ;      RETURN
1156      ;      ;(C.SWR)=DESIRED CONTROL SWITCHES
1157 035450 032777 000200 143476 GETSWR: BIT    #SW07,@SWR      ;READ CONTROL SWITCHES?
1158 035456 001430      BEQ     2$      ;NO--BRANCH
1159 035460 104401 035466      TYPE   ,65$      ;TYPE ASCII STRING
1160      035464 000410      BR      64$      ;GET OVER THE ASCII
1161      ;:65$: .ASCII <CRLF>/SET SWR<07>=0/
1162      64$:
1163 035506 012703 047264  1$:    MOV     #MSG.CS,R3      ;"CONTROL SWITCHES="
1164 035512 013704 001314      MOV     C.SWR,R4      ;PRESENT CONTROL SWITCH SETTINGS
1165 035516 004037 035542      JSR     R0,GETNUM      ;GET THE NEW SWITCH SETTINGS
1166 035522 000771      BR      1$      ; COMMA
1167 035524 000240      NOP
1168 035526 013737 001314 001320      MOV     C.SWR,SAVCSW      ;SAVE PREVIOUS VALUE
1169 035534 010437 001314      MOV     R4,C.SWR      ; DOUBLE PERIOD-SAVE NEW SWITCH SETTING
1170 035540 000207      2$:    RTS     PC      ;RETURN FROM CALL
1171
1172      ;THIS ROUTINE WILL TYPE AN ASCII MESSAGE AND THEN
1173      ;INPUT AN ASCII STRING AND CHANGE THE STRING TO OCTAL
1174      ;IF REQUIRED.

```

```

1172      ;NOTE: THIS ROUTINE DESTROYS R1
1173      ;CALL
1174      :      MOV      #ADR,R3      ;ADDRESS OF ASCII MESSAGE
1175      :      MOV      #NUM,R4      ;OCTAL NUMBER
1176      :      JSR      R0,GETNUM
1177      :      RETURN1
1178      :      RETURN2
1179      :      RETURN3
1180      :
1181      :
1182      :
1183      :
1184 035542 010337 035550      GETNUM: MOV      R3,2$      ;SAVE MESSAGE POINTER
1185 035546 104401      1$:      TYPE      ;TYPE THE MESSAGE
1186 035550 000000      2$:      .WORD      0      ;MESSAGE POINTER GOES HERE
1187 035552 010446      :      MOV      R4,-(SP)      ;SAVE R4 FOR TYPEOUT
1188 035554 104402      :      TYPOC      ;GO TYPE--OCTAL ASCII(ALL DIGITS)
1189 035556 104401 047307      :      TYPE      .SLASH      ; /
1190 035562 104411      :      RDLIN      ;READ AN ASCII STRING
1191 035564 012601      :      MOV      (SP)+,R1      ;ADDRESS OF FIRST CHARACTER
1192 035566 004037 040132      :      JSR      R0,CK.CHR      ;CHECK ONE CHARACTER
1193 035572 035546      :      1$      ;ILLEGAL CHARACTER
1194 035574 035546      :      1$      ;CARRIAGE RETURN
1195 035576 035610      :      4$      ; /
1196 035600 035634      :      8$      ;
1197 035602 035642      :      9$      ;
1198 035604 035606      :      3$      ;DIGIT 0-9
1199 035606 005301      3$:      DEC      R1      ;DECREMENT THE INPUT POINTER
1200 035610      4$:      JSR      R0,CK.NUM      ;CHECK THE NUMBER
1201 035614 035546      :      1$      ;ILLEGAL INPUT
1202 035616 035630      :      7$      ;TERMINATED WITH A '...' OR 'CR'
1203 035620 035626      :      6$      ;TERMINATED WITH A '...'
1204 035622 035624      :      5$      ;TERMINATED WITH A '...'
1205 035624 005720      5$:      TST      (R0)+      ;DOUBLE PERIOD
1206 035626 005720      6$:      TST      (R0)+      ;SINGLE PERIOD
1207 035630 010204      7$:      MOV      R2,R4      ;COMMA--SAVE INPUT NUMBER
1208 035632 000414      :      BR      11$      ;GO TO EXIT
1209 035634 105711      8$:      TSTB      (R1)      ;TERMINATOR AFTER A COMMA?
1210 035636 001343      :      BNE      1$      ;NO--LOOP
1211 035640 000411      :      BR      11$      ;YES--EXIT
1212 035642 105711      9$:      TSTB      (R1)      ;TERMINATOR AFTER A PERIOD?
1213 035644 001406      :      BEQ      10$      ;YES--EXIT
1214 035646 122721 000056      :      CMPB      #'',(R1)+      ;NO--DOUBLE PERIOD?
1215 035652 001335      :      BNE      1$      ;NO--LOOP
1216 035654 105711      :      TSTB      (R1)      ;YES--TERMINATOR?
1217 035656 001333      :      BNE      1$      ;NO--LOOP
1218 035660 005720      :      TST      (R0)+      ;DOUBLE PERIOD
1219 035662 005720      10$:      TST      (R0)+      ;PERIOD
1220 035664 010402      11$:      MOV      R4,R2      ;COMMA--POSITION THE
1221 035666 000302      :      SWAB      R2      ;NUMBER IN CASE IT
1222 035670 006202      :      ASR      R2      ;IS THE PRIORITY LEVEL
1223 035672 006202      :      ASR      R2
1224 035674 006202      :      ASR      R2
1225 035676 000200      :      RTS      R0      ;EXIT
1226
1227      ;THIS ROUTINE IS USED TO CHANGE OR MODIFY

```

```
1217 ;THE TEST PARAMETERS. IT GIVES THE OPERATOR
1218 ;THE CAPABILITY OF SPECIFYING WHICH DRIVES TO TEST, WHICH
1219 ;TESTS TO RUN AND HOW MANY TIMES TO REPEAT EACH TEST
1220
1221 035700 104412 GT.PRM: SAVREG ;SAVE R0 - R5
1222 035702 005037 001330 GT.PR1: CLR DRVSEL ;NO DRIVE SELECTED
1223 035706 104401 035714 TYPE ,65$ ;TYPE ASCIZ STRING
035712 000406 BR 64$ ;GET OVER THE ASCIZ
;65$: .ASCIZ <CRLF>/DRIVE(S)=/
64$:
1224 035730 104411 RDLIN ;READ TTY
1225 035732 012601 MOV (SP)+,R1 ;ADDRESS OF ASCIZ STRING
1226 035734 004037 040132 JSR R0,CK.CHR ;CHECK ONE CHARACTER
035740 035702 GT.PR1 ;ILLEGAL CHARACTER
035742 035702 GT.PR1 ;CARRIAGE RETURN
035744 035702 GT.PR1 ;/
035746 035702 GT.PR1 ;
035750 035702 GT.PR1 ;
035752 035754 1$ ;DIGIT 0-9
1227 035754 005301 1$: DEC R1
1228 035756 2$:
035756 012702 000007 MOV #7,R2 ;UPPER LIMIT OF INPUT
035762 004037 040206 JSR R0,CK.DIG ;CHECK THE DIGIT(S)
035766 035702 GT.PR1 ;ILLEGAL INPUT
035770 035702 GT.PR1 ;INPUT TOO LARGE
035772 036000 3$ ;TERMINATED WITH A ' ' OR 'CR'
035774 036024 4$ ;TERMINATED WITH A ' '
035776 036024 4$ ;TERMINATED WITH A ' '
1229 036000 156237 040636 001330 3$: BISB ATABIT(R2),DRVSEL ;SET THE DRIVE SELECTED BIT
1230 036006 105741 TSTB -(R1) ;WAS THE LINE TERMINATED?
1231 036010 001362 BNE 2$ ;NO-GET THE NEXT DRIVE
1232 036012 005037 001332 CLR TSTNMS ;DESELECT ALL TESTS
1233 036016 005037 001334 CLR TSTNMS+2
1234 036022 000405 BR GTTST1 ;YES--SELECT TEST
1235 036024 156237 040636 001330 4$: BISB ATABIT(R2),DRVSEL ;SET THE SELECTED DRIVE BITS
1236 036032 104413 GT.PR2: RESREG ;RESTORE R0 - R5
1237 036034 000207 RTS PC ;EXIT
1238
1239 036036 GTTST1:
036036 104401 036044 TYPE ,65$ ;TYPE ASCIZ STRING
036042 000403 BR 64$ ;GET OVER THE ASCIZ
;65$: .ASCIZ /TEST=/
64$:
1240 036052 104411 RDLIN ;READ AN ASCIZ STRING
1241 036054 012601 MOV (SP)+,R1 ;POINTER TO R1
1242 036056 122711 000056 CMPB #'.,(R1) ;DOUBLE PERIOD?
1243 036062 001007 BNE 1$ ;NO--BRANCH
1244 036064 122761 000056 000001 CMPB #'.,1(R1)
1245 036072 001003 BNE 1$
1246 036074 105761 000002 TSTB 2(R1) ;'CR'?
1247 036100 001754 BEQ GT.PR2 ;YES--EXIT
1248 036102 005037 001332 1$: CLR TSTNMS ;NO TEST SELECTED
1249 036106 005037 001334 CLR TSTNMS+2
1250 036112 005037 001336 CLR OPNFLG ;NO TESTS TO BE OPENED
1251 036116 005037 001340 CLR OPNFLG+2
1252
1253 036122 121127 000123 GTTST2: CMPB (R1),#'S ;ALL SEEK TESTS?
```

1254	036126	001004			BNE	1\$	;NO--BRANCH
1255	036130	052737	000777	001332	BIS	#777,TSTNMS	;YES--SELECT TESTS 0-10
1256	036136	000552			BR	GTTST3	
1257	036140	121127	000124		1\$: CMPB	(R1),#1	;ALL TIMING TESTS?
1258	036144	001004			BNE	2\$	;NO--BRANCH
1259	036146	052737	026000	001332	BIS	#26000,TSTNMS	;YES--SELECT TESTS 12,13 & 15
1260	036154	000543			BR	GTTST3	
1261	036156	121127	000101		2\$: CMPB	(R1),#1A	;ALL ADDRESSING TESTS?
1262	036162	001004			BNE	3\$	;NO--BRANCH
1263	036164	052737	140000	001332	BIS	#140000,TSTNMS	;YES--SELECT TESTS 16 & 17
1264	036172	000534			BR	GTTST3	
1265	036174	121127	000104		3\$: CMPB	(R1),#1D	;DATA TEST?
1266	036200	001004			BNE	4\$	;NO--BRANCH
1267	036202	052737	000001	001334	BIS	#1,TSTNMS+2	;YES--SELECT TEST 20
1268	036210	000525			BR	GTTST3	
1269	036212	121127	000105		4\$: CMPB	(R1),#1E	;EXERCISER TEST?
1270	036216	001004			BNE	5\$	;NO--BRANCH
1271	036220	052737	000002	001334	BIS	#2,TSTNMS+2	;YES--SELECT TEST 21
1272	036226	000516			BR	GTTST3	
1273	036230	004037	040056		5\$: JSR	R0,CK.OCT	;OCTAL DIGIT?
1274	036234	000514			BR	GTTST4	;NO--BRANCH
1275	036236	010205			MOV	R2,R5	;YES--SAVE IT
1276	036240	005201			INC	R1	;MOVE TO NEXT CHARACTER
1277	036242	004037	040056		JSR	R0,CK.OCT	;OCTAL DIGIT
1278	036246	000405			BR	6\$	;NO--BRANCH
1279	036250	005201			INC	R1	;MOVE TO NEXT CHARACTER
1280	036252	006305			ASL	R5	;SCALE HIGH DIGIT
1281	036254	006305			ASL	R5	
1282	036256	006305			ASL	R5	
1283	036260	060502			ADD	R5,R2	;COMBINE HIGH & LOW DIGITS
1284	036262				6\$: CMP	R2,#22	;VALID TEST NUMBER?
1287	036262	020227	000022		BGT	GTTST1	;NO--BRANCH
1288	036266	003263			MOV	R2,9\$	;SAVE THE TEST NUMBER
1289	036270	010237	036462		MOV	R2,R4	;CONVERT TEST NUMBER INTO AN INDEX
1290	036274	010204			BIC	#17,R4	;CLEAR UNWANTED BITS
1291	036276	042704	000017		ASR	R4	;SHIFT THE BITS
1292	036302	006204			ASR	R4	
1293	036304	006204			ASR	R4	
1294	036306	006204			ASL	R2	
1295	036310	006302			BIS	BITS(R2),TSTNMS(R4)	;SELECT TEST
1296	036312	056264	001540	001332	CMPB	(R1),#1-	;TEST STRING?
1297	036320	121127	000055		BNE	GTTST4	;NO--BRANCH
1298	036324	001060			INC	R1	;YES--MOVE TO NEXT CHARACTER
1299	036326	005201			JSR	R0,CK.OCT	;OCTAL DIGIT?
1300	036330	004037	040056		BR	GTTST1	;NO--BRANCH
1301	036334	000640			MOV	R2,R5	;YES--SAVE IT
1302	036336	010205			INC	R1	;MOVE TO NEXT CHARACTER
1303	036340	005201			JSR	R0,CK.OCT	;OCTAL DIGIT?
1304	036342	004037	040056		BR	7\$	;NO--BRANCH
1305	036346	000405			INC	R1	;YES--MOVE TO NEXT CHARACTER
1306	036350	005201			ASL	R5	;SCALE HIGH DIGIT
1307	036352	006305			ASL	R5	
1308	036354	006305			ASL	R5	
1309	036356	006305			ADD	R5,R2	;COMBINE HIGH & LOW DIGIT
1310	036360	060502			7\$: CMP	R2,#22	;VALID TEST NUMBER?
1311	036362						
1314	036362	020227	000022				

```
1315 036366 003223      BGT      GTTST1      ;NO--BRANCH
1316 036370 023702 036462  CMP      9$,R2      ;IS THE FIRST NUMBER OF THE
1317                      ;STRING SMALLER THAN THE LAST?
1318 036374 002220      BGE      GTTST1      ;NO--BRANCH
1319 036376 010246      MOV      R2,-(SP)    ;SAVE ENDING TEST NUMBER
1320 036400 013702 036462  MOV      9$,R2      ;GET STARTING TEST NUMBER
1321 036404 012637 036462  MOV      (SP)+,9$    ;STORE ENDING TEST NUMBER
1322 036410 006337 036462  ASL      9$      ;SHIFT ENDING TEST NUMBER
1323 036414 006302      ASL      R2      ;SHIFT TEST NUMBER
1324 036416 010204      MOV      R2,R4      ;COPY TEST NUMBER INTO R4
1325 036420 042704 000037  BIC      #37,R4    ;CLEAR LOWER BITS
1326 036424 006204      ASR      R4      ;SHIFT THE TEST NUMBER
1327 036426 006204      ASR      R4
1328 036430 006204      ASR      R4
1329 036432 006204      ASR      R4
1330 036434 056264 001540 001332  BIS      BITS(R2),TSTNMS(R4) ;SELECT THE TEST
1331 036442 062702 000002      ADD      #2,R2    ;INCREMENT THE TEST NUMBER
1332 036446 020237 036462  CMP      R2,9$    ;SEE IF FINISHED
1333 036452 101761      BLOS      8$      ;BR IF NOT
1334 036454 162702 000002      SUB      #2,R2    ;CORRECT TEST NUMBER
1335 036460 000402      BR       GTTST4      ;CONTINUE
1336 036462 000000      9$:      .WORD      0      ;STORE TEST NUMBER HERE
1337
1338 036464 005201      GTTST3: INC      R1      ;MOVE TO NEXT CHARACTER
1339 036466 121127 000056      GTTST4: CMPB     (R1),#'.    ;'PERIOD'?
1340 036472 001511      BEQ      GTTST5      ;YES--BRANCH
1341 036474 005737 001332      TST      TSTNMS    ;ANY TEST SELECTED THIS CYCLE?
1342 036500 001005      BNE      1$      ;BR IF YES
1343 036502 005737 001334      TST      TSTNMS+2  ;ANY TEST SELECTED THIS CYCLE ?
1344 036506 001002      BNE      1$      ;BR IF YES
1345 036510 000137 036036      JMP      GTTST1      ;NO
1346
1347      ;CHECK TO OPEN TESTS FOR PARAMETER CHANGES
1348
1349 036514 121127 000057      1$:      CMPB     (R1),# '/'    ;'OPEN'?
1350 036520 001054      BNE      7$      ;NO--BRANCH
1351 036522 126127 177777 000123      CMPB     -1(R1),# 'S'    ;ALL SEEK TESTS?
1352 036530 001004      BNE      2$      ;NO--BRANCH
1353 036532 052737 000777 001336      BIS      #777,OPNFLG    ;YES--OPEN TESTS 0-10
1354 036540 000451      BR       8$
1355 036542 126127 177777 000124      2$:      CMPB     -1(R1),# 'T'    ;ALL TIMING TESTS?
1356 036550 001004      BNE      3$      ;NO--BRANCH
1357 036552 052737 026000 001336      BIS      #26000,OPNFLG    ;YES--OPEN TESTS 12,13 & 15
1358 036560 000441      BR       8$
1359 036562 126127 177777 000101      3$:      CMPB     -1(R1),# 'A'    ;ALL ADDRESSING TESTS?
1360 036570 001004      BNE      4$      ;NO--BRANCH
1361 036572 052737 140000 001336      BIS      #140000,OPNFLG    ;YES--OPEN TESTS 16 & 17
1362 036600 000431      BR       8$
1363 036602 126127 177777 000104      4$:      CMPB     -1(R1),# 'D'    ;DATA TEST?
1364 036610 001004      BNE      5$      ;NO--BRANCH
1365 036612 052737 000001 001340      BIS      #1,OPNFLG+2    ;YES--OPEN TEST 20
1366 036620 000421      BR       8$
1367 036622 126127 177777 000105      5$:      CMPB     -1(R1),# 'E'    ;EXERCISER TEST?
1368 036630 001004      BNE      6$      ;NO--BRANCH
1369 036632 052737 000002 001340      BIS      #2,OPNFLG+2    ;YES--OPEN TEST 21
1370 036640 000411      BR       8$
1371 036642 056264 001540 001336      6$:      BIS      BITS(R2),OPNFLG(R4) ;YES--SET BITS FOR TEST TO OPEN
```

```
1372 036650 000405          BR      8$
1373
1374 036652 121127 000054    7$:    CMPB    (R1),#'.      ;'COMMA'?
1375 036656 001402          BEQ      8$                ;BR IF YES
1376 036660 000137 036036    JMP      GTTST1          ;NO
1377 036664 005201          8$:    INC      R1            ;MOVE TO NEXT CHARACTER
1378 036666 105711          TSTB     (R1)              ;'CR'?
1379 036670 001402          BEQ      9$                ;BR IF 'CR'
1380 036672 000137 036122    JMP      GTTST2          ;NO--GO GET NEXT CHARACTER
1381 036676 005737 001336    9$:    TST      OPNFLG        ;ANY TESTS TO OPEN ?
1382 036702 001042          BNE      OPNTST          ;BR IF YES
1383 036704 005737 001340    TST      OPNFLG+2        ;ANY TESTS TO OPEN ?
1384 036710 001037          BNE      OPNTST          ;BR IF YES
1385 036712 000137 036036    JMP      GTTST1          ;NO--START AGAIN
1386
1387 036716 005201          GTTST5: INC      R1            ;MOVE TO NEXT CHARACTER
1388 036720 121127 000056    CMPB     (R1),#'.      ;'PERIOD'?
1389 036724 001414          BEQ      GTTST6          ;YES--BRANCH
1390 036726 105711          TSTB     (R1)              ;'CR'?
1391 036730 001402          BEQ      1$                ;YES--BRANCH
1392 036732 000137 036036    JMP      GTTST1          ;ANY TESTS TO OPEN ?
1393 036736 005737 001336    1$:    TST      OPNFLG        ;BR IF YES
1394 036742 001022          BNE      OPNTST          ;ANY TESTS TO OPEN ?
1395 036744 005737 001340    TST      OPNFLG+2        ;BR IF YES
1396 036750 001017          BNE      OPNTST          ;ANY TESTS TO OPEN ?
1397 036752 000137 036032    JMP      GT.PR2          ;NO--GO START TESTING
1398
1399 036756 005201          GTTST6: INC      R1            ;MOVE TO NEXT CHARACTER
1400 036760 105711          TSTB     (R1)              ;'CR'?
1401 036762 001402          BEQ      1$                ;YES--BRANCH
1402 036764 000137 036036    JMP      GTTST1          ;NO--GO ASK FOR TEST
1403 036770 005737 001336    1$:    TST      OPNFLG        ;ANY TESTS TO OPEN ?
1404 036774 001005          BNE      OPNTST          ;BR IF YES
1405 036776 005737 001340    TST      OPNFLG+2        ;ANY TESTS TO OPEN ?
1406 037002 001002          BNE      OPNTST          ;BR IF YES
1407 037004 000137 036032    JMP      GT.PR2          ;NO--GO START TESTING
1408
1409          ;OPEN THE SELECTED TEST FOR CHANGES
1410
1411 037010 104412          OPNTST: SAVREG          ;SAVE R0 - R5
1412 037012 005027          CLR      (PC)+            ;START WITH TEST 0
1413 037014 000000          OPN.CT: .WORD    0          ;COUNT STORED HERE
1414 037016 000411          BR      OPN.2            ;SKIP THE INCREMENT
1415
1416 037020 005237 037014    OPN.1: INC      OPN.CT        ;MOVE TO THE NEXT TEST
1419 037024 022737 000022 037014    CMP      #22,OPN.CT    ;TEST NUMBER TOO BIG?
1420 037032 002003          BGE      OPN.2            ;NO--OPEN THE NEXT TEST
1421 037034 104413          RESREG          ;RESTORE R0 - R5
1422 037036 000137 036036    JMP      GTTST1          ;YES--GO ASK FOR MORE TESTS
1423
1424 037042 013705 037014    OPN.2: MOV      OPN.CT,R5        ;SETUP TO USE THE
1425 037046 006305          ASL      R5                ;TEST NUMBER AS AN INDEX
1426 037050 013703 037014    MOV      OPN.CT,R3        ;GET INDEX
1427 037054 042703 000017    BIC      #17,R3          ;CLEAR LOWER TEST BITS
1428 037060 006203          ASR      R3                ;SHIFT TEST NUMBER
1429 037062 006203          ASR      R3
1430 037064 006203          ASR      R3
```

1431	037066	036563	001540	001336	BIT	BITS(R5),OPNFLG(R3)	:OPEN THIS TEST?
1432	037074	001751			BEQ	OPN.1	:NO--MOVE TO NEXT TEST
1433	037076	104401	037104		TYPE	.65\$	:TYPE ASCII STRING
	037102	000404			BR	.64\$	:GET OVER THE ASCII
					;;65\$:	.ASCIIZ	/ TEST /
	037114				64\$:		
1434	037114	013746	037014		MOV	OPN.CT,-(SP)	:SAVE OPN.CT FOR TYPEOUT
	037120	104403			TYPOS		:GO TYPE--OCTAL ASCII
	037122	002			.BYTE	2	:TYPE 2 DIGIT(S)
	037123	000			.BYTE	0	:SUPPRESS LEADING ZEROS
1435	037124	104401	001231		TYPE	.\$CRLF	:CR-LF
1436	037130	016500	002374		MOV	PRMPT(R5),R0	:PICKUP PARAMETER POINTER
1437	037134	011046			MOV	(R0),-(SP)	:SAVE THE VARIABLE INDICATOR
1438	037136	012702	002334		MOV	#PRM,R2	:FIRST ADDRESS OF TABLE
1439	037142	000405			BR	2\$	
1440	037144	006216		1\$:	ASR	(SP)	:CHECK FOR A VARIABLE
1441	037146	103403			BCS	2\$	:GO MOVE THIS ONE
1442	037150	001404			BEQ	OPNPRM	:DONE
1443	037152	005722			TST	(R2)+	:BUMP THE POINTER
1444	037154	000773			BR	1\$	
1445	037156	012022		2\$:	MOV	(R0)+,(R2)+	:MOVE THIS VARIABLE INTO THE
1446	037160	000771			BR	1\$	:COMMON AREA
1447							
1448	037162	013716	002334		OPNPRM:	MOV	PRM,(SP)
1449	037166	005004			CLR	R4	:ZERO THE INDEX
1450	037170	006216		1\$:	ASR	(SP)	:CHECK FOR A VARIABLE
1451	037172	103403			BCS	3\$	:GO GET IT
1452	037174	001772			BEQ	OPNPRM	:OUT OF VARIABLES
1453	037176	005724		2\$:	TST	(R4)+	:UPDATE THE INDEX
1454	037200	000773			BR	1\$	
1455	037202	005764	002444	3\$:	TST	PRMLMT(R4)	:IS THE MAX. MAGNITUDE NEG?
1456	037206	100466			BMI	OPNPAT	:YES--THEN IT IS THE PATTERN
1457	037210	104401	050345		TYPE	.BLNKS2	:TYPE 2 SPACES
1458	037214	016437	002476	037224	MOV	PRMSG(R4),4\$	:TYPE THE NAME OF THIS VARIABLE
1459	037222	104401			TYPE		
1460	037224	000000		4\$:	.WORD	0	
1461	037226	104401	047262		TYPE	.MSG.EQ	:TYPE '='
1462	037232	016446	002336		MOV	RPT(R4),-(SP)	:PUT RPT(R4) ON THE STACK
	037236	004737	026136		JSR	PC,\$\$SB2D	:CHANGE TO DECIMAL ASCII
	037242	004737	026366		JSR	PC,\$\$SUPRS	:TYPE WITHOUT LEADING ZEROS
1463	037246	104401	047307		TYPE	.SLASH	:TYPE '/'
1464	037252	104411			RDLIN		
1465	037254	012601			MOV	(SP)+,R1	:READ AN ASCII STRING
1466	037256	004037	040132		JSR	R0,CK.CHR	:CHECK ONE CHARACTER
	037262	037202			3\$		:ILLEGAL CHARACTER
	037264	037176			2\$		:CARRIAGE RETURN
	037266	037334			9\$		:..
	037270	037276			5\$		:..
	037272	037304			6\$		:..
	037274	037332			8\$		:DIGIT 0-9
1467	037276	105711		5\$:	TSTB	(R1)	:..
1468	037300	001340			BNE	3\$	:NO--STAY ON THIS VARIABLE
1469	037302	000735			BR	2\$	:YES--MOVE TO NEXT VARIABLE
1470	037304	105711		6\$:	TSTB	(R1)	:IS THERE A 'CR' AFTER THE PERIOD?
1471	037306	001002			BNE	7\$	:NO
1472	037310	000137	037724		JMP	OPN.N2	:YES--GO CLOSE THIS TEST
1473	037314	122721	000056	7\$:	CMPB	#'.,(R1)+	:DOUBLE PERIOD?

1474	037320	001330		BNE	3\$		;NO--GO ASK FOR THIS VARIABLE
1475	037322	105711		TSTB	(R1)		;YES--IS A 'CR' AFTER THE DOUBLE PERIOD?
1476	037324	001326		BNE	3\$		;NO--ASK FOR THIS VARIABLE AGAIN
1477	037326	000137	037742	JMP	OPN.X2		;YES--CLOSE ALL TEST
1478							
1479	037332	005301		8\$: DEC	R1		;BACK THE POINTER UP BY ONE
1480	037334			9\$: MOV	PRMLMT(R4),R2		;UPPER LIMIT OF INPUT
	037340	016402	002444	JSR	RO,CK.DIG		;CHECK THE DIGIT(S)
	037344	004037	040206	3\$			;ILLEGAL INPUT
	037346	037202		3\$			;INPUT TO LARGE
	037350	037202		10\$			;TERMINATED WITH A '...' OR 'CR'
	037352	037356		OPN.N1			;TERMINATED WITH A '...' OR 'CR'
	037354	037720		OPN.X1			;TERMINATED WITH A '...' OR 'CR'
1481	037356	037736		10\$: MOV	R2,RPT(R4)		;SAVE THIS VARIABLE
1482	037362	010264	002336	BR	2\$		;MOVE TO NEXT VARIABLE
1483		000705					
1484							
1485							
1486	037364	104401	050345	OPNPAT: TYPE	,BLNKS2		;TYPE 2 SPACES
1487	037370	104401	047256	TYPE	,MSG.PAT		;TYPE 'PAT'
1488	037374	104401	047262	TYPE	,MSG.EQ		;TYPE '='
1489	037400	016446	002336	MOV	RPT(R4),-(SP)		;SAVE RPT(R4) FOR TYPEOUT
	037404	104402		TYPOC			;GO TYPE--OCTAL ASCII(ALL DIGITS)
1490	037406	104401	050346	TYPE	,BLNKS1		;TYPE ONE SPACE
1491	037412	104411		RDLIN			;READ ASCII STRING
1492	037414	012601		MOV	(SP)+,R1		;PICKUP POINTER
1493	037416	004037	040132	JSR	RO,CK.CHR		;CHECK ONE CHARACTER
	037422	037364		OPNPAT			;ILLEGAL CHARACTER
	037424	037162		OPNPRM			;CARRIAGE RETURN
	037426	037460		3\$			;...
	037430	037162		OPNPRM			;...
	037432	037436		1\$			;...
	037434	037456		2\$			;DIGIT 0-9
1494	037436	105711		1\$: TSTB	(R1)		;CR' AFTER THE PERIOD?
1495	037440	001531		BEQ	OPN.N2		;YES--GO CLOSE THIS TEST
1496	037442	122721	000056	CMPS	#',(R1)+		;NO--PERIOD?
1497	037446	001346		BNE	OPNPAT		;NO--LOOP
1498	037450	105711		TSTB	(R1)		;CR' AFTER A DOUBLE PERIOD?
1499	037452	001533		BEQ	OPN.X2		;YES--GO START TESTING
1500	037454	000743		BR	OPNPAT		;NO--LOOP
1501	037456	005301		2\$: DEC	R1		;BACKUP THE ASCII POINTER
1502	037460			3\$: JSR	RO,CK.NUM		;CHECK THE NUMBER
	037460	004037	040372	OPNPAT			;ILLEGAL INPUT
	037464	037364		4\$			;TERMINATED WITH A '...' OR 'CR'
	037466	037474		OPN.N1			;TERMINATED WITH A '...' OR 'CR'
	037470	037720		OPN.X1			;TERMINATED WITH A '...' OR 'CR'
	037472	037736		4\$: MOV	R2,RPT(R4)		;SAVE THE INPUT NUMBER
1503	037474	010264	002336	ROR	R2		;OPEN PATTERN 0?
1504	037500	006002		BCC	OPNPRM		;NO--START AT BEGINNING OF PARAMETER TABLE
1505	037502	103227		SAVREG			;SAVE RO - R5
1506	037504	104412					
1507							
1508							
1509							
1510	037506	005000		OPNWDS: CLR	RO		;START WITH WORD 0
1511	037510	012704	003576	MOV	#PAT0,R4		

1512	037514			1\$:			
	037514	104401	037522		TYPE	.65\$	::TYPE ASCII STRING
	037520	000403			BR	64\$	::GET OVER THE ASCII
	037530			65\$:	.ASCII	/ WD/	
1513	037530	010046		64\$:	MOV	R0,-(SP)	:PUT R0 ON THE STACK
	037532	004737	026136		JSR	PC,\$SB2D	:CHANGE TO DECIMAL ASCII
	037536	004737	026366		JSR	PC,\$SUPRS	:TYPE WITHOUT LEADING ZEROS
1514	037542	104401	047262		TYPE	.MSG.EQ	:TYPE '='
1515	037546	011446			MOV	(R4),-(SP)	:SAVE (R4) FOR TYPEOUT
	037550	104402			TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
1516	037552	104411			RDLIN		:READ ASCII STRING
1517	037554	012601			MOV	(SP)+,R1	:PICKUP THE POINTER
1518	037556	004037	040132		JSR	R0,CK.CHR	:CHECK ONE CHARACTER
	037562	037514			1\$		:ILLEGAL CHARACTER
	037564	037616			5\$		:CARRIAGE RETURN
	037566	037600			3\$		:/
	037570	037616			5\$		:...
	037572	037632			6\$		:...
	037574	037576			2\$		:DIGIT 0-9
1519	037576	005301		2\$:	DEC	R1	:BACKUP THE ASCII POINTER
1520	037600			3\$:			
	037600	004037	040372		JSR	R0,CK.NUM	:CHECK THE NUMBER
	037604	037514			1\$		:ILLEGAL INPUT
	037606	037614			4\$		:TERMINATED WITH A '...' OR '(R'
	037610	037652			7\$		:TERMINATED WITH A '...'
	037612	037664			9\$		:TERMINATED WITH A '...'
1521	037614	010214		4\$:	MOV	R2,(R4)	:SAVE THE INPUT
1522	037616	005724		5\$:	TST	(R4)+	:MOVE TO NEXT WORD
1523	037620	005200			INC	R0	:INCREMENT THE COUNT
1524	037622	022700	000020		CMP	#16.,R0	:COUNT TO LARGE?
1525	037626	003332			BGT	1\$	:NO--BRANCH
1526	037630	000726			BR	OPNWDS	:YES--BRANCH
1527	037632	105711		6\$:	TSTB	(R1)	: 'CR' AFTER THE PERIOD?
1528	037634	001407			BEQ	8\$	:YES--GO CLOSE THIS TEST
1529	037636	122721	000056		CMPB	#1.,(R1)+	:NO--PERIOD?
1530	037642	001324			BNE	1\$	:NO--BRANCH ILLEGAL INPUT STRING
1531	037644	105711			TSTB	(R1)	: 'CR' AFTER THE 'PERIOD-PERIOD'?
1532	037646	001407			BEQ	10\$	:YES--GO START TESTING
1533	037650	000721			BR	1\$	:NO--LOOP
1534	037652	010224		7\$:	MOV	R2,(R4)+	:SAVE THE INPUT
1535	037654	004737	037676	8\$:	JSR	PC,CLSWDS	:CLOSE THE DATA PATTERN
1536	037660	104413			RESREG		:RESTORE R0 - R5
1537	037662	000420			BR	OPN.N2	:MOVE TO NEXT TEST
1538	037664	010224		9\$:	MOV	R2,(R4)+	:SAVE THE INPUT
1539	037666	004737	037676	10\$:	JSR	PC,CLSWDS	:CLOSE THE DATA PATTERN
1540	037672	104413			RESREG		:RESTORE R0 - R5
1541	037674	000422			BR	OPN.X2	:START TESTING
1542							
1543							
1544							
1545	037676	012701	003576		CLSWDS:	MOV	#PAT0,R1
1546	037702	005200		1\$:	INC	R0	:FIRST ADDRESS OF DATA PATTERN
1547	037704	022700	000017		CMP	#15.,R0	:COUNT THE LAST WORD THAT WAS STORED
1548	037710	002402			BLT	2\$	:END OF TABLE
1549	037712	012124			MOV	(R1)+,(R4)+	:YES--EXIT
1550	037714	000772			BR	1\$	:COPY
							:LOOP

:CLOSE PATTERN #0 AND SAVE CHANGED WORDS

```

1551 037716 000207      2$:   RTS      PC      ;RETURN
1552
1553 037720 010264 002336  OPN.N1: MOV      R2,RPT(R4)      ;SAVE THIS VARIABLE
1554 037724 005726      OPN.N2: TST      (SP)+      ;CLEAN OFF THE STACK
1555 037726 004737 037776      JSR      PC,CLOSE      ;CLOSE THIS TEST
1556 037732 000137 037020      JMP      OPN.1      ;GO OPEN THE NEXT TEST
1557
1558 037736 010264 002336  OPN.X1: MOV      R2,RPT(R4)      ;SAVE THIS VARIABLE
1559 037742 005726      OPN.X2: TST      (SP)+      ;CLEAN OFF THE STACK
1560 037744 004737 037776      1$:   JSR      PC,CLOSE      ;CLOSE THIS TEST
1561 037750 005725      2$:   TST      (R5)+      ;UPDATE THE INDEX
1562 037752 020527 000034      CMP      R5,#16*2      ;INDEX TO BIG?
1563 037756 002403      BLT      3$      ;NO--BRANCH
1564 037760 104413      RESREG      ;RESTORE R0 - R5
1565 037762 000137 036032      JMP      GT,PR2      ;GO TO EXIT
1566 037766 036503 001540      3$:   BIT      BITS(R5),R3      ;IS THIS TEST OPEN FOR CHANGE?
1567 037772 001364      BNE      1$      ;YES--GO CLOSE IT
1568 037774 000765      BR      2$      ;NO--MOVE TO NEXT TEST
1569
1570      ;CLOSE CURRENT TEST THAT WAS OPEN FOR CHANGES
1571
1572 037776 104412      CLOSE: SAVREG      ;SAVE R0 - R5
1573 040000 012700 002334      MOV      #PRM,R0      ;'FROM' ADDRESS
1574 040004 016501 002374      MOV      PRMPT(R5),R1      ;'TO' ADDRESS
1575 040010 012002      MOV      (R0)+,R2      ;'FROM' INDICATOR
1576 040012 012103      MOV      (R1)+,R3      ;'TO' INDICATOR
1577 040014 012704 000001      MOV      #1,R4      ;TEST BIT START A 'RPT'
1578 040020 030402      1$:   BIT      R4,R2      ;PARAMETER TO BE MOVED?
1579 040022 001403      BEQ      2$      ;NO--BRANCH
1580 040024 030403      BIT      R4,R3      ;A PLACE TO PUT IT?
1581 040026 001404      BEQ      3$      ;NO--BRANCH
1582 040030 011011      MOV      (R0),(R1)      ;YES--MOVE 'FROM' TO 'TO'
1583 040032 030403      2$:   BIT      R4,R3      ;'TO' PARAMETER?
1584 040034 001401      BEQ      3$      ;NO--BRANCH
1585 040036 005721      TST      (R1)+      ;YES--UPDATE THE POINTER
1586 040040 005720      3$:   TST      (R0)+      ;UPDATE FROM POINTER
1587 040042 006304      ASL      R4      ;POSITION THE TEST BIT
1588 040044 032704 002000      BIT      #BIT10,R4      ;DONE?
1589 040050 001763      BEQ      1$      ;NO--BRANCH
1590 040052 104413      RESREG      ;RESTORE R0 - R5
1591 040054 000207      RTS      PC      ;RETURN
1592
1593      ;THIS ROUTINE IS USED TO CHECK IF AN
1594      ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
1595      ;CALL
1596      ;:   MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
1597      ;:   JSR      R0,CK.OCT      ;CHECK THE CHARACTER
1598      ;:   RETURN1      ;CHARACTER IS NOT BETWEEN 0-7
1599      ;:   RETURN2      ;CHARACTER IS IN R2 AS A
1600      ;:                   ;OCTAL DIGIT
1601
1602 040056 121127 000060      CK.OCT: CMPB      (R1),#0      ;LESS THAN ZERO?
1603 040062 103407      BLO      1$      ;YES -- BRANCH
1604 040064 121127 000067      CMPB      (R1),#7      ;GREATER THAN SEVEN?
1605 040070 101004      BHI      1$      ;YES -- BRANCH
1606 040072 111102      MOVB      (R1),R2      ;GET THE CHARACTER
1607 040074 042702 177770      BIC      #^C7,R2      ;STRIP AWAY THE ASCII

```

1608 040100 005720
 1609 040102 000200
 1610
 1611
 1612
 1613
 1614
 1615
 1616
 1617
 1618
 1619
 1620 040104 121127 000060
 1621 040110 103407
 1622 040112 121127 000071
 1623 040116 101004
 1624 040120 111102
 1625 040122 042702 000060
 1626 040126 005720
 1627 040130 000200
 1628
 1629
 1630
 1631
 1632
 1633
 1634
 1635
 1636
 1637
 1638
 1639
 1640
 1641
 1642 040132 105711
 1643 040134 001420
 1644 040136 121127 000057
 1645 040142 001414
 1646 040144 121127 000054
 1647 040150 001410
 1648 040152 121127 000056
 1649 040156 001404
 1650 040160 004037 040104
 1651 040164 000406
 1652 040166 005720
 1653 040170 005720
 1654 040172 005720
 1655 040174 005720
 1656 040176 005720
 1657 040200 005201
 1658 040202 011000
 1659 040204 000200
 1660
 1661
 1662
 1663
 1664

```

1$:   TST      (R0)+      ;ADJUST FOR RETURN
      RTS      R0         ;RETURN

;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
;CALL
;      MOV      #ADR,R1    ;ADDRESS OF ASCII CHARACTER
;      JSR      R0,CK.DEC  ;CHECK THE CHARACTER
;      RETURN1   ;NOT BETWEEN 0 AND 9
;      RETURN2   ;BETWEEN 0 AND 9
;      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      R0         ;RETURN

;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
;DETERMINE WHAT IT IS.
;CALL
;      MOV      #ADR,R1    ;ADDRESS OF ASCII CHARACTER
;      JSR      R0,CK.CHR  ;CHECK CHARACTER
;      RETURN   ADR1       ;UNKNOWN CHARACTER
;      RETURN   ADR2       ;CARRIAGE RETURN * (R1)-ADR+1
;      RETURN   ADR3       ;SLASH * (R1)=ADR+1
;      RETURN   ADR4       ;COMMA * (R1)=ADR+1
;      RETURN   ADR5       ;PERIOD * (R1)=ADR+1
;      RETURN   ADR6       ;DIGIT BETWEEN 0 AND 9.
;      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     R0,CK.DEC  ;'DIGIT'?
        BR      5$        ;NO -- BRANCH
1$:   TST      (R0)+      ;DIGIT BETWEEN 0-9
2$:   TST      (R0)+      ;PERIOD
3$:   TST      (R0)+      ;COMMA
4$:   TST      (R0)+      ;SLASH
5$:   INC      R1         ;CARRIAGE RETURN
      MOV      (R0),R0    ;MOVE POINTER TO NEXT CHARACTER
      RTS      R0         ;UNKNOWN CHARACTER
      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

```

1665			:	MOV	#NUM,R2	:	MAX. MAGNITUDE OF INPUT NUMBER
1666			:	JSR	RO,CK.DIG	:	CHECK DIGITS
1667			:	RETURN	ADR1	:	ILLEGAL CHARACTER -- R2-?
1668			:	RETURN	ADR2	:	INPUT NUMBER TOO LARGE -- R2-?
1669			:	RETURN	ADR3	:	'COMMA' -- R2 = NUMBER
1670			:	RETURN	ADR4	:	'PERIOD' -- R2 = NUMBER
1671			:	RETURN	ADR5	:	'PERIOD-PERIOD' -- R2 = NUMBER
1672			:				
1673	040206	010446		CK.DIG: MOV	R4,-(SP)	:	SAVE R4
1674	040210	010346		MOV	R3,-(SP)	:	SAVE R3
1675	040212	010246		MOV	R2,-(SP)	:	SAVE THE MAX. SIZE ON THE STACK
1676	040214	005002		CLR	R2	:	START WITH 0
1677	040216	005003		CLR	R3		
1678	040220	005004		CLR	R4		
1679	040222	004037	040132	JSR	RO,CK.CHR	:	CHECK ONE CHARACTER
	040226	040356		9\$		:	ILLEGAL CHARACTER
	040230	040356		9\$		:	CARRIAGE RETURN
	040232	040356		9\$		:	...
	040234	040356		9\$		:	...
	040236	040356		9\$		:	...
	040240	040242		1\$		:	DIGIT 0-9
1680	040242	006303		ASL	R3	:	2
1681	040244	010346		MOV	R3,-(SP)	:	SAVE *2
1682	040246	006303		ASL	R3	:	4
1683	040250	006303		ASL	R3	:	8
1684	040252	062603		ADD	(SP)+,R3	:	(*8)+(*2)=*10.
1685	040254	060203		ADD	R2,R3	:	UPDATE THE INPUT NUMBER
1686	040256	004037	040132	JSR	RO,CK.CHR	:	CHECK ONE CHARACTER
	040262	040356		9\$		:	ILLEGAL CHARACTER
	040264	040276		2\$		:	CARRIAGE RETURN
	040266	040356		9\$		:	...
	040270	040304		4\$		:	...
	040272	040302		3\$		:	...
	040274	040242		1\$		:	DIGIT 0-9
1687	040276	005301		DEC	R1	:	BACKUP THE CHARACTER POINTER
1688	040300	000401		BR	4\$	:	CONTINUE
1689	040302	005724		3\$: TST	(R4)+	:	'PERIOD'
1690	040304	005724		4\$: TST	(R4)+	:	'COMMA' OR 'CR'
1691	040306	004037	040132	JSR	RO,CK.CHR	:	CHECK ONE CHARACTER
	040312	040356		9\$		:	ILLEGAL CHARACTER
	040314	040346		7\$		:	CARRIAGE RETURN
	040316	040356		9\$		:	...
	040320	040356		9\$		:	...
	040322	040326		5\$		:	...
	040324	040336		6\$		:	DIGIT 0-9
1692	040326	005724		5\$: TST	(R4)+	:	'PERIOD-PERIOD'
1693	040330	105711		TSTB	(R1)	:	'CR'?
1694	040332	001405		BEQ	7\$	:	YES--BRANCH
1695	040334	000410		BR	9\$		
1696	040336	126127	177776 000054	6\$: CMPB	-2(R1),#'	:	WAS CHARACTER BEFORE THE DIGIT A 'COMMA'?
1697	040344	001004		BNE	9\$	:	NO--EXIT
1698	040346	020316		7\$: CMP	R3,(SP)	:	INPUT TOO LARGE?
1699	040350	101001		BHI	8\$	:	YES -- BRANCH
1700	040352	060400		ADD	R4,R0	:	ADJUST RETURN ADDRESS
1701	040354	005720		8\$: TST	(R0)+		
1702	040356	010302		9\$: MOV	R3,R2	:	NUMBER TO R2
1703	040360	005726		TST	(SP)+	:	CLEAN MAX. SIZE OFF OF STACK

```

1704 040362 012603      MOV      (SP)+,R3      ;RESTORE R3
1705 040364 012604      MOV      (SP)+,R4      ;RESTORE R4
1706 040366 011000      MOV      (R0),R0      ;GET RETURN ADDRESS
1707 040370 000200      RTS      R0          ;RETURN
1708
1709                      ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL CHARACTERS
1710                      ;AND FORMS AN OCTAL NUMBER IN R2
1711                      ;CALL:
1712                      ;      MOV      #ADR,R1      ;ADDRESS OF ASCII STRING
1713                      ;      JSR      R0,CK.NUM    ;GO FORM THE NUMBER
1714                      ;      RETURN  ADR1        ;ILLEGAL CHARACTER IN THE INPUT STRING
1715                      ;      RETURN  ADR2        ;'COMMA' OR 'CR'--R2=NUMBER
1716                      ;      RETURN  ADR3        ;'PERIOD'--R2=NUMBER
1717                      ;      RETURN  ADR4        ;'PERIOD-PERIOD'--R2=NUMBER
1718
1719 040372 010346      CK.NUM: MOV      R3,-(SP)      ;SAVE R3
1720 040374 005003      CLR      R3              ;START NUMBER AT ZERO
1721 040376 004037 040056 JSR      R0,CK.OCT      ;OCTAL DIGIT?
1722 040402 000440      BR      6$              ;NO--BRANCH
1723 040404 005201      1$: INC      R1          ;MOVE TO NEXT CHARACTER
1724 040406 006303      ASL      R3              ;FOR THE OCTAL NUMBER IN R3
1725 040410 103435      BCS      6$              ;DON'T LET IT GET TO BIG
1726 040412 006303      ASL      R3
1727 040414 103433      BCS      6$
1728 040416 006303      ASL      R3
1729 040420 103431      BCS      6$
1730 040422 060203      ADD      R2,R3
1731 040424 004037 040056 JSR      R0,CK.OCT      ;IS THIS AN OCTAL DIGIT?
1732 040430 000401      BR      2$              ;NO--FIND OUT WHAT IT IS
1733 040432 000764      BR      1$              ;YES--MAKE IT PART OF THE NUMBER
1734 040434 010302      2$: MOV      R3,R2      ;SAVE THE OCTAL NUMBER
1735 040436 005003      CLR      R3              ;START WITH ZERO INDEX
1736 040440 004037 040132 JSR      R0,CK.CHR      ;CHECK ONE CHARACTER
1737 040444 040504      6$                      ;ILLEGAL CHARACTER
1738 040446 040474      5$                      ;CARRIAGE RETURN
1739 040450 040504      6$                      ;'/'
1740 040452 040474      5$                      ;'.'
1741 040454 040460      3$                      ;'0'-'9'
1742 040456 040504      6$                      ;DIGIT 0-9
1743 040460 005723      3$: TST      (R3)+      ;'PERIOD'
1744 040462 121127 000056 CMPB    (R1),#'.    ;'PERIOD-PERIOD'?
1745 040466 001002      BNE      5$              ;NO--BRANCH
1746 040470 005201      INC      R1          ;YES--ADVANCE THE POINTER
1747 040472 005723      4$: TST      (R3)+      ;'PERIOD-PERIOD'
1748 040474 005723      5$: TST      (R3)+      ;'COMMA'
1749 040476 105711      TSTB    (R1)          ;'CR'?
1750 040500 001001      BNE      6$              ;NO--BRANCH
1751 040502 060300      ADD      R3,R0          ;YES--SAVE THE OCTAL NUMBER
1752 040504 012603      6$: MOV      (SP)+,R3      ;RESTORE R3
1753 040506 011000      MOV      (R0),R0      ;PICKUP EXIT ADDRESS
1754 040510 000200      RTS      R0          ;RETURN

```

```

1      .SBTTL  SINGLE/DUAL PORT RH/RM DRIVER (REV 6.5) 1981
2
3      ;NEW DRIVE TYPE ID FOR RM02 *****
4      ;10-AUG-77 *****
5      ;10-MAR-78 THE SC, SC5 CHANGES
6      ;NEW DRIVE TYPE ID FOR RM05 *****
7      ;1980 *****
8
9      ;COPYRIGHT (C) 1977,1981
10     ;DIGITAL EQUIPMENT CORP.
11     ;MAYNARD, MA 01754
12     ;AUTHOR(S): JIM LACEY/CHUCK HESS
13     ;REVISED BY: MIKE LEAVITT      11-APR-80, 27-MAR-81
14
15     ;*****
16
17     ;STORAGE FOR RMDS, RMER1, RMER2, AND RMMR2 ON AN ERROR '2'
18     ;RMERRS  = RMDS
19     ;RMERRS+2 = RMER1
20     ;RMERRS+4 = RMER2
21     ;RMERRS+6 = RMMR2
22
23 040512 000000 000000 000000 RMERRS: .WORD 0,0,0,0
24
25     ;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)
26     ;DRVACT=0 IF DRIVE IS IDLE
27     ;DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
28     ;DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION
29
30 040522      000      DRVACT: .BYTE 0      ;DRIVE 0
31 040523      000      .BYTE 0      ;DRIVE 1
32 040524      000      .BYTE 0      ;DRIVE 2
33 040525      000      .BYTE 0      ;DRIVE 3
34 040526      000      .BYTE 0      ;DRIVE 4
35 040527      000      .BYTE 0      ;DRIVE 5
36 040530      000      .BYTE 0      ;DRIVE 6
37 040531      000      .BYTE 0      ;DRIVE 7
38
39     ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
40     ;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXISTENT
41     ;DRVSTA>0 IF DRIVE IS ONLINE
42     ;DRVSTA<0 IF DRIVE IS UNSAFE
43
44 040532      000      DRVSTA: .BYTE 0      ;DRIVE 0
45 040533      000      .BYTE 0      ;DRIVE 1
46 040534      000      .BYTE 0      ;DRIVE 2
47 040535      000      .BYTE 0      ;DRIVE 3
48 040536      000      .BYTE 0      ;DRIVE 4
49 040537      000      .BYTE 0      ;DRIVE 5
50 040540      000      .BYTE 0      ;DRIVE 6
51 040541      000      .BYTE 0      ;DRIVE 7
52
53     ;TABLE OF DRIVE TYPES (DRV Typ=8 BYTES)
54     ;DRV Typ=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
55     ;DRV Typ=7 IF DRIVE IS RM05 *****
56     ;DRV Typ=5 IF DRIVE IS RM02 *****
57     ;DRV Typ=4 IF DRIVE IS RM03
  
```

```
50                                     ;DRVTYPE=-1 IF NOT RM05/3/2
51
52 040542      000      DRVTYPE: .BYTE 0      ;DRIVE 0
55 040543      000      .BYTE 0      ;DRIVE 1
    040544      000      .BYTE 0      ;DRIVE 2
    040545      000      .BYTE 0      ;DRIVE 3
    040546      000      .BYTE 0      ;DRIVE 4
    040547      000      .BYTE 0      ;DRIVE 5
    040550      000      .BYTE 0      ;DRIVE 6
    040551      000      .BYTE 0      ;DRIVE 7
56
57                                     ;TABLE OF DUAL PORT INITIALIZATION INDICATORS
58                                     ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
59                                     ;DPINT<0 IF INITIALIZATION IS IN PROGRESS
60
61 040552      000      DPINT: .BYTE 0      ;DRIVE 0
64 040553      000      .BYTE 0      ;DRIVE 1
    040554      000      .BYTE 0      ;DRIVE 2
    040555      000      .BYTE 0      ;DRIVE 3
    040556      000      .BYTE 0      ;DRIVE 4
    040557      000      .BYTE 0      ;DRIVE 5
    040560      000      .BYTE 0      ;DRIVE 6
    040561      000      .BYTE 0      ;DRIVE 7
65
66                                     ;TABLE OF PENDING DUAL PORT REQUESTS
67                                     ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
68                                     ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE
69
70 040562      000      DPRQS: .BYTE 0      ;DRIVE 0
73 040563      000      .BYTE 0      ;DRIVE 1
    040564      000      .BYTE 0      ;DRIVE 2
    040565      000      .BYTE 0      ;DRIVE 3
    040566      000      .BYTE 0      ;DRIVE 4
    040567      000      .BYTE 0      ;DRIVE 5
    040570      000      .BYTE 0      ;DRIVE 6
    040571      000      .BYTE 0      ;DRIVE 7
74
75                                     ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
76                                     ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
77                                     ;'DPB' OF THE I/O OPERATION.
78
79 040572      000000    TRNSWT: .WORD 0
80
81                                     ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
82                                     ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
83                                     ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
84                                     ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
85                                     ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
86
87 040574      000000    SRCHWT: .WORD 0
88
89                                     ;RM DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
90                                     ;ACTDRV=0 IF DRIVER IS INACTIVE
91                                     ;ACTDRV>0 IF DRIVER IS ACTIVE
92
93 040576      000      ACTDRV: .BYTE 0
94
```

```

95      ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
96      ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
97      ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
98
99 040577      000      ACTSTR: .BYTE      0
100
101      ;UNLOAD FLAG (ULDFLG=8 BYTES)
102      ;ULDFLG=0 IF NO UNLOAD COMMAND
103      ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
104      ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE
105
106 040600      000      ULDFLG: .BYTE      0      ;DRIVE 0
107 040601      000      .BYTE      0      ;DRIVE 1
108 040602      000      .BYTE      0      ;DRIVE 2
109 040603      000      .BYTE      0      ;DRIVE 3
110 040604      000      .BYTE      0      ;DRIVE 4
111 040605      000      .BYTE      0      ;DRIVE 5
112 040606      000      .BYTE      0      ;DRIVE 6
113 040607      000      .BYTE      0      ;DRIVE 7
114
115      ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
116      ;SAVEFG <0 IF SAVE THE RH/RM REGISTERS WHEN THE
117      ;OPERATION IS COMPLETED AS PER (DPB+14).
118      ;SAVEFG=0 IF SAVE THE RH/RM REGISTERS, AS PER
119      ;(DPB+14), AFTER AN ERROR.
120
121 040610      000000      SAVEFG: .WORD      0
122
123      ;SEEK FLAG (SEEKFG=1 WORD)
124      ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
125      ;FOR A DATA TRANSFER START A SEARCH COMMAND
126      ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
127      ;DISREGARD THE WINDOW
128
129 040612      177777      SEEKFG: .WORD     -1
130
131      ;TIMEOUT TABLE (TIMER=8 WORDS)
132      ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
133
134 040614      177777      TIMER:   .WORD     -1      ;DRIVE 0
135 040616      177777      .WORD     -1      ;DRIVE 1
136 040620      177777      .WORD     -1      ;DRIVE 2
137 040622      177777      .WORD     -1      ;DRIVE 3
138 040624      177777      .WORD     -1      ;DRIVE 4
139 040626      177777      .WORD     -1      ;DRIVE 5
140 040630      177777      .WORD     -1      ;DRIVE 6
141 040632      177777      .WORD     -1      ;DRIVE 7
142
143      ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
144      ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
145      ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
146
147 040634      177777      DTUW:    .WORD     -1
148
149      ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
150      ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
151      ;ATTENTION BIT

```

```

144
145 040636      001      ATABIT: .BYTE 1      ;DRIVE 0
146 040637      002      .BYTE 2      ;DRIVE 1
147 040640      004      .BYTE 4      ;DRIVE 2
148 040641      010      .BYTE 10     ;DRIVE 3
149 040642      020      .BYTE 20     ;DRIVE 4
150 040643      040      .BYTE 40     ;DRIVE 5
151 040644      100      .BYTE 100    ;DRIVE 6
152 040645      200      .BYTE 200    ;DRIVE 7
153
154      ;NUMBER OF 'MASSBUS CONTROL PARITY ERRORS' (MCPE) ALLOWED BEFORE
155      ;CALLING IT FATAL (MCPEMX=1 WORD)
156
157 040646      000003    MCPEMX: .WORD 3
158
159      ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH/RM),
160      ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).
161
162 040650      176700    RMADR: .WORD 176700
163 040652      000254    000240    RMVEC: .WORD 254,5*32.
164
165      ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
166 040656      000005    MXWNDW: .WORD 5
167
168      ;DEFINITIONS OF THE RH/RM ADDRESS INDEXES
169
170
171
172
173      000000    RMCS1 = 0      ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 0)
174      000002    RMWC = 2      ;WORD COUNT REGISTER (NOT A DRIVE REG)
175      000004    RMBA = 4      ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
176      000006    RMDA = 6      ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 5)
177      000010    RMCS2 = 10     ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
178      000012    RMDS = 12     ;DRIVE STATUS REGISTER (DRIVE REG 1)
179      000014    RMER1 = 14     ;ERROR REGISTER #1 (DRIVE REG. 2)
180      000016    RMAS = 16     ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 4)
181      000020    RMLA = 20     ;LOOK AHEAD REGISTER (DRIVE REG. 7)
182      000022    RMDB = 22     ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
183      000024    RMMR1 = 24     ;MAINTAINABILITY REGISTER (DRIVE REG. 3)
184      000026    RMDT = 26     ;DRIVE TYPE REGISTER (DRIVE REG. 6)
185      000030    RMSN = 30     ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
186      000032    RMOF = 32     ;OFFSET REGISTER (DRIVE REG. 11)
187      000034    RMDC = 34     ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
188      000036    RMHR = 36     ;DUMMY ADDRESS REGISTER (DRIVE REG. 13)
189      000040    RMMR2 = 40     ;MAINTENANCE REGISTER #2
190      000042    RMER2 = 42     ;ERROR REGISTER #2 (DRIVE REG. 15)
191      000044    RMEC1 = 44     ;ECC POSITION REGISTER (DRIVE REG. 16)
192      000046    RMEC2 = 46     ;ECC PATTERN REGISTER (DRIVE REG. 17)
193
194      .SBTTL RH/RM DRIVER INITIALIZATION CODE
195
196      ;THIS ROUTINE WILL DETERMINE WHICH RM DRIVES ARE
197      ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
198      ;TO THE PROPER STATE FOR EACH DRIVE.
199      ;NOTE: THIS ROUTINE CALLS DRVINT
200
201      ;
202      ;
203      ;CALL
204      ;
205      ;
206      JSR      PC,RMINIT
207

```

```

208      :      RETURN
209      :
210      :NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
211      :
212      RMINIT: SAVREG      ;SAVE R0 - R5
213      MOV      PS,-(SP)    ;SAVE THE PRESENT PROCESSOR STATUS
214      MOV      #<5*32.>,PS ;CHANGE THE PRIORITY TO 5
215      JSR      PC,CLRQUE   ;CLEAR ALL REQUEST QUEUES
216      MOV      #RMERRS,R1  ;FIRST ADDRESS TO BE CLEARED
217      MOV      #SEKFG,R2   ;LAST ADDRESS TO BE CLEARED
218      1$:      CLR      (R1)+ ;CLEAR
219      CMP      R1,R2        ;ARE WE DONE?
220      BLO      1$           ;BR IF NO
221      MOV      #DTUW,R2     ;LAST ADDRESS
222      2$:      MOV      #-1,(R1)+ ;INITIALIZE
223      CMP      R1,R2        ;DONE?
224      BLOS     2$           ;LOOP IF NO
225      CLR      DRVSTA       ;SET ALL DRIVES TO OFFLINE
226      CLR      DRVSTA+2
227      CLR      DRVSTA+4
228      CLR      DRVSTA+6
229      MOV      RMVEC,R3     ;SETUP THE RH/RM VECTOR
230      MOV      #ISR,(R3)+
231      MOV      RMVEC+2,(R3)
232      MOV      RMADR,R4     ;FIRST ADDRESS OF RH/RM
233      MOV      #CLR,RMCS2(R4) ;MASSBUS INIT
234      CLR      R1           ;START WITH DRIVE 0
235      3$:      JSR      R0,DRVINT ;INIT THE DRIVE
236      BR      4$           ;'DVA' NOT SET OR PARITY ERROR
237      BR      5$           ;NORMAL RETURN
238      4$:      CLRB      DRVSTA(R1) ;SET DRIVE STATUS TO OFFLINE
239      5$:      INC      R1       ;GO TO NEXT DRIVE
240      BIC      #^C7,R1      ;MASK OUT UNUSED BITS
241      BNE      3$           ;BR IF MORE DRIVES TO GO
242      MOV      #7,R1        ;START WITH DRIVE 7
243      CLR      PS           ;CLEAR THE PROCESSOR STATUS
244      6$:      TSTB      DPINT(R1) ;WAITING FOR DRIVE TO SWITCH PORTS ?
245      BEQ      8$           ;BR NOT WAITING
246      JSR      PC,SET.IE    ;SET INTERRUPT
247      7$:      TSTB      DPINT(R1) ;DRIVE SWITCHED PORTS ?
248      BNE      7$           ;BR IF NOT
249      8$:      DEC      R1       ;GO TO THE NEXT DRIVE
250      BPL      6$           ;CHECK NEXT DRIVE
251      MOV      (SP)+,PS      ;RESTORE THE PROCESSOR STATUS
252      RESREG      PC         ;RESTORE R0 - R5
253      RTS
254      :
255      :DRIVE INITIALIZATION ROUTINE
256      :THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
257      :AN RM05/3/2. IF IT IS, A 'READ-IN PRESET' IS ISSUED AND FMT16
258      :IS SET TO A '1'. THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
259      :INSURE THEY ARE ALL ON A '1'. AND DEPENDING ON THEIR STATE,
260      :DRVSTA IS SET TO THE PROPER CONDITION.
261      :CALL
262      :      MOV      #DRVNUM,R1 ;DRIVE NUMBER TO R1
263      :      MOV      RMADR,R4   ;UNIBUS ADDRESS OF RH/RM (RMCS1)
264      :      JSR      R0,DRVINT  ;CALLED BY A JSR
  
```

```

265      :      RETURN1      :ERROR OCCURRED (PARITY)
266      :      RETURN2      :NORMAL RETURN
267      :
268      :
269 041072 010546      DRVINT: MOV      R5,-(SP)      :SAVE R5
270 041074 105061 040532      CLRB      DRVSTA(R1)      :START DRIVE STATUS AS OFFLINE
271 041100 105061 040542      CLRB      DRVSTYP(R1)      :CLEAR THE DRIVE TYPE INDICATOR
272 041104 105061 040600      CLRB      ULDFLG(R1)      :CLEAR THE UNLOAD FLAG
273 041110 010164 000010      MOV      R1,RMCS2(R4)      :SELECT A DRIVE
274 041114 112764 000111 000000      MOV      #11,RMCS1(R4)      :DO A DRIVE CLEAR COMMAND (& SIZE DRIVE)
275 041122 032764 010000 000010      BIT      #BIT12,RMCS2(R4)      :NONEXISTENT DRIVE?
276 041130 001403      BEQ      1$      :NO
277 041132 004737 045772      JSR      PC,SET.IE      :GO SET 'IE' WITHOUT A 'TRE'
278 041136 000520      BR      4$      :LEAVE THIS ROUTINE
279
280 041140 105061 040532      1$:      CLRB      DRVSTA(R1)      :SET DRIVE STATUS TO OFFLINE
281 041144 032764 004000 000000      BIT      #BIT11,RMCS1(R4)      :SEE IF DRIVE AVAILABLE
282 041152 001512      BEQ      4$      :BR IF DRIVE NOT AVAILABLE
283 041154 004037 045302      JSR      R0,RD.RM      :READ THE DRIVE TYPE REG.
284 041160 000026      RMDT      5$
285 041162 041402      5$      :ERROR RETURN ADDRESS
286 041164 012605      MOV      (SP)+,R5      :PUT DRIVE TYPE IN R5
287 041166 112761 000004 040542      MOV      #4,DRVSTYP(R1)      :SET RM03 INDICATOR
288 041174 022705 020024      CMP      #20024,R5      :SINGLE PORT RM03 ?
289 041200 001431      BEQ      2$      :BR IF YES
290 041202 022705 024024      CMP      #24024,R5      :DUAL PORT RM03 ?
291 041206 001426      BEQ      2$      :BR IF YES
292 041210 112761 000005 040542      MOV      #5,DRVSTYP(R1)      :SET RM02 INDICATOR
293 041216 022705 020025      CMP      #20025,R5      :SINGLE PORT RM02 ?
294 041222 001420      BEQ      2$      :BR IF SO
295 041224 022705 024025      CMP      #24025,R5      :DUAL PORT RM02 ?
296 041230 001415      BEQ      2$      :BR IF SO
297 041232 112761 000007 040542      MOV      #7,DRVSTYP(R1)      :SET RM05 INDICATOR
298 041240 022705 020027      CMP      #20027,R5      :SINGLE PORT RM05 ?
299 041244 001407      BEQ      2$      :BR IF YES
300 041246 022705 024027      CMP      #24027,R5      :DUAL PORT RM05 ?
301 041252 001404      BEQ      2$      :BR IF YES
302 041254 112761 177777 040542      MOV      #-1,DRVSTYP(R1)      :SET INDICATOR TO 'OTHER'
303 041262 000446      BR      4$      :EXIT
304
305 041264 012746 000121      2$:      MOV      #121,-(SP)      :DO A 'READ-IN PRESET'
306 041270 004037 045462      JSR      R0,WRT.RM
307 041274 000000      RMCS1
308 041276 041402      5$
309 041300 012746 010000      MOV      #BIT12,-(SP)      :SET FMT16=1
310 041304 004037 045462      JSR      R0,WRT.RM
311 041310 000032      RMOF
312 041312 041402      5$
313 041314 004037 045302      JSR      R0,RD.RM      :READ RMD5
314 041320 000012      RMD5
315 041322 041402      5$
316 041324 012605      MOV      (SP)+,R5      :AND SAVE IT IN R5
317 041326 100015      BPL      3$      :BR IF ATA=0
318 041330 116164 040636 000016      MOV      ATABIT(R1),RMAS(R4)      :CLEAR ATTENTION BIT
319 041336 004037 045302      JSR      R0,RD.RM      :FIND OUT WHY ATA-1
320 041342 000014      RMER1
321 041344 041402      5$

```

```

322 041346 006126          ROL      (SP)+      ;IS IT UNSAFE?
323 041350 100004          BPL      3$      ;BR IF NOT
324 041352 112761 177777 040532  MOVB   #-1,DRVSTA(R1) ;SET UNSAFE INDICATOR
325 041360 000407          BR       4$      ;EXIT
326
327 041362 005105          3$:  COM     R5      ;CHECK MOL, DPR, DRY, AND VV
328 041364 042705 167077    BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
329 041370 001003          BNE     4$      ;BR IF MOL, DPR, DRY, OR VV IS CLEAR
330 041372 112761 000001 040532  MOVB   #1,DRVSTA(R1) ;SET DRIVE STATUS TO ONLINE
331 041400 005720          4$:  TST     (R0)+    ;STEP OVER THE ERROR RETURN
332 041402 012605          5$:  MOV     (SP)+,R5 ;RESTORE R5
333 041404 000200          RTS     R0      ;EXIT
334
335          ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
336
337          ;CALL
338
339          JSR     R0,RM05      ;CALL THE RM05 DRIVER
340          PNTADR  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
341          RETURN1 ;RETURN HERE IF QUEUE IS FULL
342          RETURN2 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
343          ;IS AN ERROR CONDITION
344
345 041406 013746 177776  RM05:  MOV     PS,-(SP) ;SAVE THE CALLING STATUS
346 041412 013737 040654 177776  MOV     RMVEC+2,PS ;DON'T ALLOW ANY RM INTERRUPTS
347 041420 112737 000001 040576  MOVB   #1,ACTDRV ;SET 'ACTIVE DRIVER' FLAG
348 041426 104412          SAVREG ;SAVE R0 - R5
349 041430 011002          MOV     (R0),R2 ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
350 041432 005062 000016          CLR     16(R2) ;CLEAR THE STATUS/ERROR INDICATOR
351 041436 111201          MOVB   (R2),R1 ;PICKUP THE DRIVE NUMBER
352 041440 013704 040650          MOV     RMADR,R4 ;UNIBUS ADDRESS OF RMCS1
353 041444 105761 040532          TSTB   DRVSTA(R1) ;CHECK DRIVES STATUS
354 041450 003014          BGT     1$      ;BR IF ONLINE
355 041452 105761 040600          TSTB   ULDFLG(R1) ;UNLOAD COMMAND IN QUEUE?
356 041456 001036          BNE     3$      ;BR IF YES
357 041460 105761 040552          TSTB   DPINT(R1) ;TRYING TO INIT THE DRIVE
358 041464 001042          BNE     5$      ;BR IF YES
359 041466 004037 041072          JSR     R0,DRVINT ;GO INIT. THE DRIVE
360 041472 000434          BR       4$      ;ERROR RETURN
361 041474 105761 040532          TSTB   DRVSTA(R1) ;IS DRIVE STATUS ONLINE?
362 041500 003445          BLE     6$      ;BR IF NOT
363 041502 105761 040562          1$:  TSTB   DPRQS(R1) ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
364 041506 001031          BNE     5$      ;BR IF YES
365 041510 010164 000010          MOV     R1,RMCS2(R4) ;SELECT THE DRIVE
366 041514 004037 046434          JSR     R0,DRVQUE ;PUT THIS REQUEST IN QUEUE
367 041520 000460          BR       9$      ;QUEUE IS FULL
368 041522 122762 000103 000002  CMPB   #103,2(R2) ;IS THIS REQ. FOR AN UNLOAD?
369 041530 001003          BNE     2$      ;BR IF NO
370 041532 112761 177777 040600  MOVB   #-1,ULDFLG(R1) ;SET THE 'UNLOAD IN QUEUE' FLAG
371 041540 105761 040522          2$:  TSTB   DRVACT(R1) ;IS THIS DRIVE ACTIVE?
372 041544 001043          BNE     8$      ;BR IF YES
373 041546 004737 041700          JSR     PC,OPT ;CALL THE OPTIMIZER
374 041552 000440          BR       8$      ;
375 041554 012762 120000 000016  3$:  MOV     #BIT15!BIT13,16(R2) ;SET THE 'UNLOAD IN QUEUE' ERROR FLAG
376 041562 000434          BR       8$      ;EXIT
377 041564 004737 042756          4$:  JSR     PC,C17 ;GO HANDLE THE PARITY ERROR
378 041570 000431          BR       8$

```

```

379 041572 004037 046434 5$: JSR R0,DRVQUE ;PUT REQUEST IN QUEUE
380 041576 000431 9$: BR 9$ ;QUEUE IS FULL
381 041600 032714 000100 BIT #BIT06,(R4) ;IE BIT SET?
382 041604 001023 BNE 8$ ;YES
383 041606 004737 045772 JSR PC,SET.IE ;SET THE INTERRUPT
384 041612 000420 BR 8$ ;RETURN
385 041614 105761 040532 6$: TSTB DRVSTA(R1) ;SEE IF DRIVE OFFLINE OR UNSAFE
386 041620 002412 BLT 7$ ;BR IF UNSAFE
387 041622 012762 140000 000016 MOV #BIT15:BIT14,16(R2) ;SET OFFLINE ERROR INDICATOR
388 041630 105761 040542 TSTB DRVSTYP(R1) ;SEE IF OFFLINE OR NONEXISTENT
389 041634 001007 BNE 8$ ;BR IF OFFLINE
390 041636 012762 100002 000016 MOV #BIT15:BIT01,16(R2) ;REPORT DRIVE NONEXISTENT
391 041644 000403 BR 8$ ;GO TO EXIT
392 041646 012762 110000 000016 7$: MOV #BIT15:BIT12,16(R2) ;DRIVE IS UNSAFE
393 041654 104413 8$: RESREG ;RESTORE R0 - R5
394 041656 005720 TST (R0)+ ;SETUP FOR NORMAL RETURN
395 041660 000401 BR 10$ ;FINISH UP, THEN EXIT
396 041662 104413 9$: RESREG ;RESTORE R0 - R5
397 041664 005720 10$: TST (R0)+ ;CORRECT THE RETURN ADDRESS
398 041666 105037 040576 CLR B ACTDRV ;CLEAR 'ACTIVE DRIVER' FLAG
399 041672 012637 177776 MOV (SP)+,PS ;RETURN 'PS' TO USER LEVEL
400 041676 000200 RTS R0 ;RETURN TO CALLER
401
402 ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
403
404 ;CALL
405
406 MOV #DRVNUM,R1 ;DRIVE NUMBER TO R1
407 JSR PC,OPT ;SETUP A COMMAND
408
409 OPT: SAVREG ;SAVE R0 - R5
410 MOV PS,-(SP) ;SAVE PROC. STATUS
411 BICB ATABIT(R1),SRCHWT ;CLEAR LA SEACH FLAG
412 CLRB DPRQS(R1) ;RESET THE PORT REQ FLAG ****
413 JSR PC,GETREQ ;GET 'DPR' POINTER OF REQUEST
414 TST R2 ;IS THERE A REQUEST IN QUEUE?
415 BEQ 7$ ;NO--BR TO EXIT
416 MOV R1,RMCS2(R4) ;LOAD THE DRIVE ADDRESS *****
417 MOV #111,RMCS1(R4) ;CLEAR THE DRIVE
418 BIT #BIT11,RMCS1(R4) ;DVA SET?
419 BEQ 5$ ;TO PORT REQUEST, IF NOT
420 TSTB DRVSTA(R1) ;IS DRIVE ONLINE?
421 BGT 2$ ;YES
422 JSR PC,POPQUE ;NO--REMOVE REQUEST FROM QUEUE
423 MOV #BIT15:BIT14,16(R2) ;SET OFFLINE STATUS/ERROR INDICATOR
424 TSTB DRVSTA(R1) ;IS DRIVE UNSAFE?
425 BPL 8$ ;BR TO EXIT IF NOT
426 MOV #BIT15:BIT12,16(R2) ;SET UNSAFE STATUS/ERROR INDICATOR
427 BR 8$ ;BR TO EXIT
428
429 2$: CMPB #150,2(R2) ;IS THE REQUEST FOR I/O?
430 BLT 3$ ;YES
431 JSR PC,C14 ;CALL THE COMMAND INITIATOR
432 BR 8$ ;BR TO EXIT
433
434 3$: TST DTUW ;DATA TRANSFER UNDERWAY?
435 BGE 4$ ;YES--GO START A SEARCH
436 TST SEEKFG ;DO IMPLIED SEEKS?
437 BPL 4$ ;NO, DO A SEARCH

```

```

444 042042 004737 042126      JSR    PC,C11      ;START A DATA TRANSFER
445 042046 000423              BR      8$
446 042050 004737 042234      JSR    PC,C13      ;START A SEARCH
447 042054 000420              BR      8$      ;GO TO THE EXIT
448 042056 112761 177777 040562 5$:  MOVB   #-1,DPRQS(R1) ;SET PORT REQUEST INDICATOR
449 042064 010103              MOV    R1,R3      ;SET UP TO ADDRESS WORDS
450 042066 006303              ASL     R3          ;CONVERT TO WORD INDEX
451 042070 012763 035230 040614  MOV    #15000.,TIMER(R3) ;START 15. SECOND TIMER
452 042076 000402              BR      7$      ;EXIT
453 042100 004737 042756      6$:  JSR    PC,C17      ;PROCESS THE PARITY ERROR
454 042104 032714 000100      7$:  BIT    #BIT06,(R4) ;SEE IF 'IE' ALREADY SET
455 042110 001002              BNE     8$      ;BR IF SET
456 042112 004737 045772      JSR    PC,SET.IE    ;SET 'IE' WITHOUT A 'TRE'
457 042116 012637 177776      8$:  MOV    (SP)+,PS    ;RESTORE PROC. STATUS
458 042122 104413              RESREG ;RESTORE R0 - R5
459 042124 000207              RTS     PC
460
461      ;COMMAND INITIATOR
462
463      ;CALL
464      :
465      :      MOV    #DRVNUM,R1      ;DRIVE NUMBER
466      :      MOV    #DPB,R2        ;ADDRESS OF DPB
467      :      JSR    PC,C1?         ;C1?= C11,C13, OR C14
468      :      :
469      :      :      ;WHERE:
470      :      :      :      ;C11=DATA TRANSFER
471      :      :      :      ;C13=SEARCH REQUESTED BY DATA XFER
472      :      :      :      ;C14=NOT DATA TRANSFER
473
472 042126 004737 046532      C11:  JSR    PC,POPQUE    ;REMOVE REQUEST FROM 'DRIVES WAIT' QUEUE
473 042132 010237 040572      MOV    R2,TRNSWT ;PUT REQ. IN TRANSFER WAIT QUEUE
474 042136 010203              MOV    R2,R3      ;DPB ADDRESS TO R3
475 042140 013704 040650      MOV    RMADR,R4    ;RMCS1 ADDRESS
476 042144 010164 000010      MOV    R1,RMCS2(R4) ;SELECT DRIVE
477 042150 062703 000004      ADD     #4,R3      ;DESIRED WORD COUNT
478 042154 062704 000002      ADD     #2,R4      ;RMWC ADDRESS
479 042160 012324              MOV    (R3)+,(R4)+ ;LOAD WORD COUNT
480 042162 012324              MOV    (R3)+,(R4)+ ;LOAD BUFFER ADDRESS
481 042164 012346              MOV    (R3)+,-(SP) ;LOAD SECTOR AND TRACK
482 042166 004037 045462      JSR    R0,WRT.RM    ;CALL THE LOAD(WRITE) ROUTINE
483 042172 000006              RMDA     C17      ;INDEX OF REGISTER TO LOAD
484 042174 042756              MOV    (R3)+,-(SP) ;ERROR RETURN ADDRESS
485 042176 012346              MOV    (R3)+,-(SP) ;LOAD CYLINDER ADDRESS
486 042200 004037 045462      JSR    R0,WRT.RM
487 042204 000034              RMDC     C17
488 042206 042756              MOV    2(R2),-(SP) ;LOAD 'COMMAND+GO', 'A17&A16', AND 'PSEL'
489 042210 016246 000002      JSR    R0,WRT.RM
490 042214 004037 045462      RMDC     C17
491 042220 000000              MOV    R1,DTUW    ;SET 'DATA TRANSFER UNDERWAY'
492 042222 042756              JMP     C15
493 042224 010137 040634      C13:  MOV    RMADR,R4    ;RMCS1 ADDRESS
494 042230 000137 042720      MOV    R1,RMCS2(R4) ;SELECT DRIVE
495
496 042234 013704 040650      MOV    12(R2),-(SP) ;DESIRED CYLINDER ADDRESS
497 042240 010164 000010      JSR    R0,WRT.RM
498 042244 016246 000012      RMDC
499 042250 004037 045462
500 042254 000034

```

501	042256	042756			C17		
502	042260	116203	000010		MOVB	10(R2),R3	;PICKUP SECTOR ADDRESS
503	042264	163703	040656		SUB	MXWINDW,R3	;BACKUP BY MAX. SEARCH FOR I/O WINDOW
504	042270	002002			BGE	1\$	
505	042272	062703	000040		ADD	#32.,R3	
506	042276	010346		1\$:	MOV	R3,-(SP)	;COMBINE THE ADJUSTED SECTOR WITH
507	042300	116266	000011	000001	MOVB	11(R2),1(SP)	;THE DESIRED TRACK
508	042306	004037	045462		JSR	R0,WRT.RM	;LOAD DESIRED TRACK & SECTOR
509	042312	000006			RMDA		
510	042314	042756			C17		
511	042316	012746	000131		MOV	#131,-(SP)	;START A SEARCH
512	042322	004037	045462		JSR	R0,WRT.RM	
513	042326	000000			RMCS1		
514	042330	042756			C17		
515	042332	156137	040636	040574	BISB	ATAB1,(R1),SRCHWT	;SET 'SEARCH WAIT' KEY
516	042340	000567			BR	C15	
517							
518	042342	013704	040650		MOV	RMADR,R4	;RMCS1 ADDRESS
519	042346	010164	000010		MOV	R1,RMCS2(R4)	;SELECT DRIVE
520	042352	116203	000002		MOVB	2(R2),R3	;PICKUP THE REQUESTED COMMAND
521	042356	122703	000131		CMPS	#131,R3	;IS IT A SEARCH COMMAND?
522	042362	001007			BNE	1\$	;BR IF NO
523	042364	016246	000010		MOV	10(R2),-(SP)	;LOAD DESIRED TRACK & SECTOR
524	042370	004037	045462		JSR	R0,WRT.RM	
525	042374	000006			RMDA		
526	042376	042756			C17		
527	042400	000403			BR	2\$	;GO LOAD CYLINDER
528	042402	122703	000105	1\$:	CMPS	#105,R3	;IS IT A SEEK COMMAND
529	042406	001007			BNE	3\$	;BR IF NO
530	042410	016246	000012	2\$:	MOV	12(R2),-(SP)	;LOAD DESIRED CYLINDER
531	042414	004037	045462		JSR	R0,WRT.RM	
532	042420	000034			RMDC		
533	042422	042756			C17		
534	042424	000546			BR	C16	
535	042426	122703	000115	3\$:	CMPS	#115,R3	;IS IT AN 'OFFSET' COMMAND?
536	042432	001013			BNE	4\$	;BR IF NO
537	042434	004037	045302		JSR	R0,RD.RM	;MERGE THE OFFSET VALUE INTO RMOF
538	042440	000032			RMOF		;BUT DON'T CHANGE THE UPPER
539	042442	042756			C17		
540	042444	116216	000001		MOVB	1(R2),(SP)	;BYTE WHEN LOADING THE
541	042450	004037	045462		JSR	R0,WRT.RM	;REGISTER (RMOF)
542	042454	000032			RMOF		
543	042456	042756			C17		
544	042460	000530			BR	C16	;GO START THE COMMAND
545	042462	122703	000107	4\$:	CMPS	#107,R3	;IS IT A 'RECALIBRATE' COMMAND?
546	042466	001525			BEQ	C16	;BR IF YES
547	042470	122703	000117		CMPS	#117,R3	;IS IT A RETURN TO CENTER?
548	042474	001522			BEQ	C16	;BR IF YES
549	042476	122703	000103		CMPS	#103,R3	;IS IT AN 'UNLOAD' COMMAND?
550	042502	001016			BNE	5\$	;BR IF NO
551	042504	112761	000001	040522	MOVB	#1,DRVACT(R1)	;SET THE DRIVE ACTIVE INDICATOR
552	042512	105061	040532		CLRB	DRVSTA(R1)	;PUT DRIVE STATUS TO OFFLINE
553	042516	112761	000001	040600	MOVB	#1,ULDFLG(R1)	;SET 'UNLOAD IN PROGRESS' FLAG
554	042524	010346			MOV	R3,-(SP)	;START THE 'UNLOAD' COMMAND
555	042526	004037	045462		JSR	R0,WRT.RM	
556	042532	000000			RMCS1		
557	042534	042756			C17		

558	042536	000207			RTS	PC	:RETURN TO USER
559	042540	122703	000143	5\$:	CMPB	#143,R3	:IS IT A 'SET FORMAT' COMMAND?
560	042544	001014			BNE	6\$	:BR IF NO
561	042546	004037	045302		JSR	R0,RD.RM	:READ THE OFFSET REGISTER
562	042552	000032			RMOC		
563	042554	042756			CI7		
564	042556	116266	000001	000001	MOVB	1(R2),1(SP)	:COMBINE 'FMT16','ECI', AND 'HCI'
565	042564	004037	045462		JSR	R0,WRT.RM	:LOAD 'FMT16','ECI', AND/OR 'HCI'.
566	042570	000032			RMOC		
567	042572	042756			CI7		
568	042574	000436			BR	12\$	
569	042576	122703	000141	6\$:	CMPB	#141,R3	:IS IT A 'GET REGISTER' COMMAND?
570	042602	001023			BNE	10\$	:BR IF NO
571	042604	016203	000006	7\$:	MOV	6(R2),R3	:POINTS TO 1ST ADDRESS OF WHERE
572							:TO PUT THE REGISTER(S)
573	042610	116237	000010	042626	MOVB	10(R2),9\$	:INIT. THE INDEX FOR THE FIRST REG.
574	042616	116205	000011		MOVB	11(R2),R5	:INDEX OF LAST REG. TO MOVE
575	042622	004037	045302	8\$:	JSR	R0,RD.RM	:READ RH/RM REGISTER
576	042626	000000		9\$:	RMCS1		:INDEX OF REG. TO READ
577	042630	042756			CI7		
578	042632	012623			MOV	(SP)+,(R3)+	:GET THE CONTENTS OF RH/RM REG.
579	042634	023705	042626		CMP	9\$,R5	:LAST REG. BEEN READ?
580	042640	001414			BEQ	12\$	:GET OUT IF YES
581	042642	062737	000002	042626	ADD	#2,9\$	:INCREASE THE INDEX BY 2
582	042650	000764			BR	8\$	:LOOP--MORE TO READ
583	042652	122703	000145	10\$:	CMPB	#145,R3	:IS IT A 'SELECT DRIVE' COMMAND?
584	042656	001405			BEQ	12\$	:BR IF YES
585	042660	010346		11\$:	MOV	R3,-(SP)	:LOAD THE COMMAND
586	042662	004037	045462		JSR	R0,WRT.RM	
587	042666	000000			RMCS1		
588	042670	042756			CI7		
589	042672	004737	046532	12\$:	JSR	PC,POPQUE	:REMOVE REQ. FROM QUEUE
590	042676	052762	000200	000016	BIS	#BIT07,16(R2)	:SET THE 'DONE' BIT
591	042704	005737	040610		TST	SAVEFG	:SAVE THE RH/RM REGISTERS?
592	042710	100002			BPL	13\$	:BR IF NO
593	042712	004737	045654		JSR	PC,SVRH70	:YES--GO SAVE THE REGISTERS
594	042716	000207		13\$:	RTS	PC	:RETURN TO USER
595							
596	042720	006301			ASL	R1	
597	042722	012761	023420	040614	MOV	#10000.,TIMER(R1)	:START 10. SECOND TIMER
598	042730	006201			ASR	R1	
599	042732	112761	000001	040522	MOVB	#1,DRVACT(R1)	:SET THE DRIVE ACTIVE
600	042740	000207			RTS	PC	:RETURN TO THE USER
601							
602	042742	010346			MOV	R3,-(SP)	:LOAD THE COMMAND
603	042744	004037	045462		JSR	R0,WRT.RM	
604	042750	000000			RMCS1		
605	042752	042756			CI7		
606	042754	000761			BR	CI5	
607							
608	042756	032764	010000	000010	BIT	#BIT12,RMCS2(R4)	:DRIVE NON-EXISTENT ?
612	042764	005702		1\$:	TST	R2	:ANYTHING IN QUEUE ?
616	042766	001001			BNE	2\$	:BR IF QUEUE IS THERE
617	042770	000207			RTS	PC	:OTHERWISE EXIT
618	042772	012762	104000	000016	MOV	#BIT15,BIT11,16(R2)	:SET 'PARITY' ERROR INDICATOR
622							
623	043000	012746	000111		MOV	#111,-(SP)	:DO A 'DRIVE CLEAR'
				CI7B:			

```

624 043004 004037 045462 JSR R0,WRT,RM
625 043010 000000 RMCS1
626 043012 043056 C18
627 043014 004737 046414 1$: JSR PC,EMPTYQ ;EMPTY THE QUEUE
628 043020 105061 040562 CLR B DPRQS(R1) ;CLEAR THE PORT REQUEST FLAG
629 043024 105061 040600 CLR B ULDFLG(R1) ;CLEAR THE UNLOAD IN QUEUE FLAG
630 043030 105061 040522 CLR B DRVACT(R1) ;DRIVE IS IDLE
631 043034 020237 040572 CMP R2,TRNSWT ;IF THIS DRIVE HAD AN I/O REQUEST
635 043040 001005 BNE 2$ ;IN PROGRESS CLEAR ALL OF THE FLAGS
636 043042 005037 040572 CLR TRNSWT
637 043046 012737 177777 040634 MOV #1,DTUW
638 043054 000207 2$: RTS PC
639
640 043056 104412 C18: SAVREG ;SAVE R0 - R5
641 043060 005001 CLR R1
642 043062 005003 CLR R3
643 043064 105761 040522 1$: TST B DRVACT(R1) ;DRIVE ACTIVE?
644 043070 001003 BNE 2$ ;BR IF IN ACTIVE
645 043072 105761 040562 TST B DPRQS(R1) ;PORT REQUEST
646 043076 001443 BEQ 6$ ;BR IF NOT
647 043100 013702 040572 2$: MOV TRNSWT,R2 ;GET THE 'TRANSFER WAIT' QUEUE
648 043104 020137 040634 CMP R1,DTUW ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
649 043110 001402 BEQ 3$ ;BR IF YES
650 043112 004737 046510 JSR PC,GETREQ ;GET THE DPB POINTER
651 043116 005702 3$: TST R2 ;QUEUE ENTRY FOR DRIVE ?
652 043120 001413 BEQ 5$ ;BR IF NOT
653 043122 032764 010000 000010 BIT #BIT12,RMCS2(R4) ;'NED' SET ?
654 043130 001404 BEQ 4$ ;BR IF NOT
655 043132 012762 100002 000016 MOV #BIT15:BIT01,16(R2) ;SET 'DRIVE NON-EXISTENT' INDICATOR
656 043140 000403 BR 5$ ;CONTINUE
657 043142 012762 102000 000016 4$: MOV #BIT15:BIT10,16(R2) ;SET 'NON-CLEARABLE PARITY' ERROR INDICATOR
661 043150 012763 177777 040614 5$: MOV #1,TIMER(R3) ;STOP THE TIMER
662 043156 105061 040522 CLR B DRVACT(R1) ;SET 'DRIVE ACTIVE' TO IDLE
663 043162 105061 040562 CLR B DPRQS(R1) ;CLEAR PORT REQUEST FLAG
664 043166 020137 040634 CMP R1,DTUW ;IS THIS DRIVE SETUP FOR A TRANSFER
665 043172 001005 BNE 6$ ;BR IF NOT
666 043174 012737 177777 040634 MOV #1,DTUW ;RESET THE INDICATOR
667 043202 005037 040572 CLR TRNSWT ;CLEAR THE TRANSFER QUEUE
668 043206 105061 040600 6$: CLR B ULDFLG(R1) ;CLEAR UNLOAD FLAG
669 043212 032764 010000 000010 BIT #BIT12,RMCS2(R4) ;'NED' SET ?
673 043220 005201 INC R1 ;MOVE TO THE NEXT DRIVE
674 043222 062703 000002 ADD #2,R3
675 043226 042701 177770 BIC #C7,R1
676 043232 001314 BNE 1$ ;BR IF MORE DRIVES
677 043234 012737 177777 040634 MOV #1,DTUW ;NO DATA TRANSFERS UNDERWAY
678 043242 005037 040572 CLR TRNSWT ;CLEAR THE 'TRANSFER WAIT' QUEUE
679 043246 004737 046336 JSR PC,CLRQUE ;CLEAR ALL OF THE REQUEST QUEUES
680 043252 012764 000040 C00010 MOV #CLR,RMCS2(R4) ;DO A MASSBUS INIT.
681 043260 000406 BR 8$ ;CONTINUE
682 043262 004737 046414 7$: JSR PC,EMPTYQ ;CLEAR THE DRIVE'S QUEUE
683 043266 105061 040532 CLR B DRVSTA(R1) ;SET DRIVE TO OFFLINE
684 043272 105061 040542 CLR B DRVTYP(R1) ;CLEAR THE DRIVE TYPE INDICATOR
685 043276 004737 045772 8$: JSR PC,SET.IE ;SET 'IE' WITHOUT 'TRE'
686 043302 104413 RESREG ;RESTORE R0 - R5
687 043304 000207 RTS PC ;RETURN
688
689 ;INTERRUPT SERVICE ROUTINE

```

```

690
691 043306 112737 000001 040576 ISR:  MOVB  #1,ACTDRV      ;SET 'ACTIVE DRIVER' FLAG
692 043314 104412          SAVREG          ;SAVE R0 - R5
693 043316 013704 040650          MOV  RMADR,R4      ;ADDRESS OF RMCS1
694 043322 013701 040634          MOV  DTUW,R1      ;GET 'DATA TRANSFER UNDERWAY' INDICATOR
695 043326 002402          BLT  1$              ;BR IF NO DATA TRANSFER UNDERWAY
696 043330 004737 043350          JSR  PC,TD        ;CALL TRANSFER DONE
697 043334 004737 043520          1$: JSR  PC,SC      ;CALL SPECIAL CONDITIONS
698 043340 104413          2$:  RESREG          ;RESTORE R0 - R5
699 043342 105037 040576          CLRB  ACTDRV      ;CLEAR 'ACTIVE DRIVER' FLAG
700 043346 000002          RTI                    ;RETURN
701
702          ;TRANSFER DONE ROUTINE
703
704 043350 105061 040522          TD:  CLRB  DRVACT(R1)  ;SET DRIVE ACTIVE INDICATOR TO IDLE
705 043354 012737 177777 040634          MOV  #-1,DTUW      ;NO DATA TRANSFERS UNDERWAY
706 043362 006301          ASL  R1
707 043364 012761 177777 040614          MOV  #-1,TIMER(R1) ;CANCEL TIMEOUT
708 043372 006201          ASR  R1
709 043374 013702 040572          MOV  TRNSWT,R2      ;GET 'DPB' ADDRESS FROM THE
710 043400 005037 040572          CLR  TRNSWT      ;TRANSFER WAIT QUEUE--CLEAR QUEUE
711 043404 052762 000200 000016          BIS  #BIT07,16(R2) ;SET DONE
712 043412 010164 000010          MOV  R1,RMCS2(R4) ;SELECT THE DRIVE
713 043416 004037 045302          JSR  R0,RD.RM      ;TRANSFER ERROR(TRE=1)?
714 043422 000000          RMCS1
715 043424 042756          C17
716 043426 006126          ROL  (SP)+              ;IS TRE=1 ?
717 043430 100417          BMI  3$              ;BR IF YES
718 043432 005737 040610          TST  SAVEFG      ;SAVE THE RH/RM REGISTERS?
719 043436 100002          BPL  1$              ;BR IF NO
720 043440 004737 045654          JSR  PC,SVRH70     ;YES--SAVE THE REGISTERS
721 043444 004737 046510          1$: JSR  PC,GETREQ   ;GET DPB POINTER
722 043450 005702          TST  R2              ;ENTRY FOR DRIVE ?
723 043452 001403          BEQ  2$              ;BR IF NOT
724 043454 004737 041700          JSR  PC,OPT        ;CALL OPTIMIZER
725 043460 000207          RTS  PC              ;RETURN
726 043462 012714 000113          2$: MOV  #113,(R4)   ;RELEASE THE DRIVE
727 043466 000207          RTS  PC              ;RETURN
728
729 043470 052762 100100 000016 3$: BIS  #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
730 043476 004737 046414          JSR  PC,EMPTYQ     ;EMPTY THE 'DRIVE'S WAIT' QUEUE
731 043502 004737 045654          JSR  PC,SVRH70     ;SAVE THE RH/RM REGISTERS
732 043506 012714 040111          MOV  #40111,(R4)   ;ISSUE A 'DRIVE CLEAR'
733 043512 012714 000113          MOV  #113,(R4)   ;ISSUE A RELEASE TO THE DRIVE
734 043516 000207          RTS  PC              ;RETURN
735
736          ;SPECIAL CONDITION ROUTINE
737
738 043520 116403 000016          SC:  MOVB  RMAS(R4),R3 ;READ 'RMAS'
739 043524 001014          BNE  2$              ;BR IF ANY 'ATA' BITS SET
740 043526 004037 045302          JSR  R0,RD.RM      ;READ CONTROL AND STATUS REGISTER
741 043532 000000          RMCS1
742 043534 043056          C18
743 043536 106126          ROLB (SP)+              ;IS 'IE'=1?
744 043540 100405          BMI  1$              ;YES, NO DRIVES TO CHECK
745 043542 004037 046600          JSR  R0,ES.SAV     ;SAVE THE ADDRESS IN '$ESCAPE'
746 043546 104001          EMT  1              ;REPORT AN ILLEGAL INTERRUPT

```

```

773 043550 004737 045772      JSR      PC,SET.IE      ;SET INTERRUPT ENABLE
774 043554 000207      RTS      PC              ;RETURN
775 043556 005046      CLR      -(SP)          ;PROCESS ALL DRIVES THAT HAVE
776 043560 110316      MOV      R3,(SP)        ;AN 'ATA'=1
777 043562 012703 000001      MOV      #1,R3
778 043566 005001      CLR      R1
779
780 043570 030316      SC3:    BIT      R3,(SP)      ;ATA=1?
781 043572 001005      BNE      SC5              ;YES
782
783 043574 005201      SC4:    INC      R1              ;MOVE TO THE NEXT DRIVE
784 043576 106303      ASLB      R3
785 043600 001373      BNE      SC3              ;BR IF MORE TO CHECK?
786 043602 005726      TST      (SP)+          ;CLEAN OFF THE STACK
787 043604 000207      RTS      PC              ;RETURN TO USER
788
789 043606 105761 040552      SC5:    TSTB     DPINT(R1)    ;INITIALIZING THE DRIVE ?
790 043612 001402      BEQ      1$              ;BR IF NOT
791 043614 000137 044532      JMP      SC13          ;PROCESS THE DRIVE
792 043620 105761 040562      1$:    TSTB     DPRQS(R1)    ;PORT REQUEST OUTSTANDING ?
793 043624 001402      BEQ      2$              ;BR IF NOT
794 043626 000137 044532      JMP      SC13          ;START THE OUTSTANDING COMMAND
795 043632 105761 040532      2$:    TSTB     DRVSTA(R1)    ;CHECK THE DRIVE STATUS
796 043636 003023      BGT      4$              ;BR IF ONLINE
797 043640 105761 040600      TSTB     ULDFLG(R1)    ;UNLOAD IN PROGRESS?
798 043644 003420      BLE      4$              ;BR IF NOT
799 043646 004737 046510      JSR      PC,GETREQ      ;GET DPB POINTER
800 043652 004737 045654      JSR      PC,SVRH70      ;SAVE THE RH/RM REGISTERS
801 043656 004737 044462      JSR      PC,SC12      ;SAVE RMD5, RMER1, RMER2, AND RMMR2
802
803 043662 105761 040532      TSTB     DRVSTA(R1)    ;ALSO DO A DRIVE INIT (DRVINT)
804 043666 003414      BLE      5$              ;DID DRIVE COME ONLINE?
805 043670 032737 040000 040512      BIT      #BIT14,RMERRS    ;NO
806 043676 001000      BNE      3$              ;WAS THERE AN ERROR?
807 043678 001000      BNE      3$              ;BR IF ERROR
808 043680 001000      BNE      3$              ;BR IF ERROR
809 043682 001000      BNE      3$              ;BR IF ERROR
810 043700 013705 040514      3$:    MOV      RMERRS+2,R5    ;YES -- PICKUP RMER1 AND
811 043704 000504      BR       SC6A          ;GO PROCESS THE ERROR
812 043706 105761 040522      4$:    TSTB     DRVACT(R1)    ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
813 043712 001033      BNE      SC6              ;BR IF EITHER
814 043714 004737 044462      JSR      PC,SC12      ;SAVE RMD5, RMER1, RMER2, AND RMMR2
815
816 043720 105761 040552      5$:    TSTB     DPINT(R1)    ;ALSO DO A DRVINT
817 043724 001323      BNE      SC4              ;TRYING TO INIT THE DRIVE ?
818 043726 105761 040532      TSTB     DRVSTA(R1)    ;BR IF YES, CHECK ON MORE DRIVES
819 043732 100412      BMI      6$              ;CHECK ON DRIVE'S STATUS
820 043734 032737 020000 040516      BIT      #BIT13,RMERRS+4    ;BR IF UNSAFE
821 043742 001013      BNE      7$              ;ADDRESS PLUG CHANGED ?
822 043744 012746 000111      MOV      #111,-(SP)    ;BR IF YES
823 043746 004037 045462      JSR      R0,WRT.RM    ;DRIVE CLEAR
824 043748 000000      RMCS1      ;WRITE THE COMMAND INTO RMCS1
825 043750 004037 045462      SC8          ;REGISTER INDEX
826 043752 004322      SC8          ;PARITY EXIT ADDRESS
827 043754 011605      6$:    MOV      (SP),R5    ;PICKUP (RMAS) BEFORE THE ERROR CALL
828 043756 004037 046600      JSR      R0,ES.SAV    ;SAVE THE ADDRESS IN '$ESCAPE'
829 043758 010402      EMT      2              ;REPORT THE UNEXPECTED ATTENTION
830 043760 000701      BR       SC4              ;GO CHECK FOR MORE ATA'S
831 043770 004037 046600      7$:    JSR      R0,ES.SAV    ;SAVE THE ADDRESS IN '$ESCAPE'
832 043772 010405      EMT      5              ;REPORT THE ADDRESS PLUG CHANGE
833 043774 010405

```

```

833 044000 000675      BR      SC4      ;CHECK FOR MORE DRIVES
834
835 044002 006301      SC6:  ASL      R1      ;SETUP TO ADDRESS WORDS
836 044004 012761 177777 040614  MOV    #-1,TIMER(R1) ;STOP THE TIMER
837 044012 006201      ASR      R1      ;RESTORE THE DRIVE ADDRESS
838 044014 004737 046510  JSR      PC,GETREQ ;GET THE DPB POINTER FROM THE QUEUE
839 044020 010164 000010  MOV    R1,RMCS2(R4) ;SELECT DRIVE
840 044024 000137 044352  JMP      SC11 ;PROCESS THE SEARCH
841 044030 004037 045302  JSR      R0,RD.RM ;READ THE RM'S STATUS REG.
842 044034 000012      RMDS
843 044036 044322      SC8
844 044040 011605      MOV    (SP),R5 ;AND PUT IT IN R5
845 044042 006126      ROL      (SP)+ ;WAS THERE AN ERROR?
846 044044 100407      BMI      1$ ;BR IF ERROR
847 044046 105761 040522  TSTB    DRVACT(R1) ;CHECK DRIVE'S STATE
848 044052 003137      BGT      SC11 ;BR IF DRIVE ACTIVE WITH ORDER
849 044054 052762 100210 000016  BIS    #BIT15!BIT07!BIT03,16(R2) ;INFORM USER OF ERROR RECOVER COMPLETION
850 044062 000470      BR      SC7
851 044064 004037 045302  1$:  JSR      R0,RD.RM ;READ ERROR REGISTER #1
852 044070 000014      RMER1
853 044072 044322      SC8
854 044074 012605      MOV    (SP)+,R5 ;AND SAVE IT IN R5
855 044076 004737 045654  JSR      PC,SVRH70 -- ;SAVE RM/RM REGISTERS
856 044102 012746 000111  MOV    #111,-(SP) ;ISSUE A DRIVE CLEAR
857 044106 004037 045462  JSR      R0,WRT.RM
858 044112 000000      RMCS1
859 044114 044322      SC8
860
861 044116 006105      SC6A: ROL      R5 ;WAS 'UNSAFE' CONDITION -1?
862 044120 100406      BMI      1$ ;BR IF YES
863 044122 005702      TST      R2 ;ANYTHING IN QUEUE ?
864 044124 001447      BEQ      SC7 ;BR IF NOT
865 044126 052762 100240 000016  BIS    #BIT15!BIT07!BIT05,16(R2) ;INFORM USER OF ERROR
866 044134 000443      BR      SC7
867 044136 004037 045302  1$:  JSR      R0,RD.RM ;READ DRIVE STATUS REG. #1
868 044142 000012      RMDS
869 044144 044322      SC8
870 044146 011605      MOV    (SP),R5 ;SAVE RMDS IN R5
871 044150 006126      ROL      (SP)+ ;'ERR'=1?
872 044152 100011      BPL      2$ ;BR IF NO--UNSAFE CLEARED
873 044154 112761 177777 040532  MOVB   #-1,DRVSTA(R1) ;DRIVE IS UNSAFE
874 044162 004737 045654  JSR      PC,SVRH70 ;SAVE RM/RM REGISTERS
875 044166 052762 110000 000016  BIS    #BIT15!BIT12,16(R2) ;INFORM USER OF UNSAFE ERROR
876 044174 000423      BR      SC7
877 044176 032705 010000  2$:  BIT      #BIT12,R5 ;'MOL' = 1 ?
878 044202 001015      BNE      3$ ;BR IF YES
879 044204 112761 177777 040522  MOVB   #-1,DRVACT(R1) ;ACTIVE ERROR RECOVER
880 044212 112761 000001 040532  MOVB   #1,DRVSTA(R1) ;ONLINE
881 044220 006301      ASL      R1
882 044222 012761 035230 040614  MOV    #15000.,TIMER(R1) ;START 15. SECOND TIMER
883 044230 006201      ASR      R1
884 044232 000137 043574  JMP      SC4
885 044236 052762 100220 000016  3$:  BIS    #BIT15!BIT07!BIT04,16(R2) ;INFORM USER OF ERROR
886
887 044244 105061 040522  SC7:  CLRB    DRVACT(R1) ;DRIVE IS IDLE
888 044250 004737 046532  JSR      PC,POPQUE ;REMOVE THE QUEUE
889 044254 105761 040600  TSTB    ULDFLG(R1) ;UNLOAD IN PROGRESS OR QUEUE?

```

```

893 044260 003002      BGT      1$      ;BR IF NOT
894 044262 105061 040600 CLR      ULDFLG(R1)      ;CLEAR UNLOAD FLAG
895 044266 116164 040636 000016 1$: MOV     ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
896 044274 105761 040532      TSTB     DRVSTA(R1)      ;IS THE DRIVE UNSAFE ?
897 044300 100406      BMI      2$      ;BR IF IT IS
901 044302 012746 000111      MOV     #111,-(SP)      ;DRIVE CLEAR COMMAND
902 044306 004037 045462      JSR      RO,WRT.RM      ;WRITE THE COMMAND INTO RPCS1
903 044312 000000      RMCS1     ;REGISTER INDEX
904 044314 044322      SC8       ;PARITY EXIT ADDRESS
905 044316 000137 043574      2$: JMP      SC4       ;CHECK FOR MORE DRIVES
906
907 044322 105761 040522      SC8: TSTB     DRVACT(R1)      ;IS DRIVE IDLE?
908 044326 001405      BEQ      1$      ;YES
909 044330 004737 046510      JSR      PC,GETREQ      ;GET DPB POINTER
910 044334 004737 042756      JSR      PC,C17       ;PROCESS THE PARITY ERROR
911 044340 000402      BR       2$      ;CONTINUE
912 044342      1$:
916 044342 004737 043000      JSR      PC,C17B      ;PROCESS THE UNCORRECTABLE PARITY ERROR
917 044346 000137 043574      2$: JMP      SC4       ;CHECK MORE DRIVES
918
919 044352 105761 040600      SC11: TSTB     ULDFLG(R1)      ;'UNLOAD IN PROGRESS'?
920 044356 003402      BLE      1$      ;BR IF NO
921 044360 105061 040600      CLR      ULDFLG(R1)      ;CLEAR UNLOAD FLAG
922 044364 105061 040522      1$: CLR      DRVACT(R1)      ;SET DRIVE IDLE
923 044370 136137 040636 040574      BITB     ATABIT(R1),SRCHWT ;DOING A SEARCH OPERATION FOR
924                                     ;AN I/O COMMAND?
925 044376 001012      BNE      2$      ;BR IF YES
926 044400 004737 046532      JSR      PC,POPQUE      ;REMOVE REQUEST FROM QUEUE
927 044404 052762 000200 000016      BIS      #BIT07,16(R2) ;SET 'DONE' BIT
928 044412 005737 040610      TST      SAVEFG      ;SAVE THE REGISTERS?
929 044416 100002      BPL      2$      ;BR IF NO
930 044420 004737 045654      JSR      PC,SVRH70      ;YES--SAVE ALL OF THE RH/RM REG'S
931 044424 116164 040636 000016 2$: MOV     ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
932 044432 146137 040636 040574      BICB     ATABIT(R1),SRCHWT ;CLEAR IMPLIED SEEK SET
933 044440 006301      ASL      R1       ;WORD INDEX
934 044442 012761 177777 040614      MOV     #-1,TIMER(R1) ;STOP CLOCK
935 044450 006201      ASR      R1       ;RESTORE R1
936 044452 004737 041700      JSR      PC,OPT       ;START A REQUEST
937 044456 000137 043574      JMP      SC4       ;CHECK FOR MORE DRIVES
938
939 044462 010164 000010      SC12: MOV     R1,RMCS2(R4) ;SELECT DRIVE
940 044466 016437 000012 040512      MOV     RMD5(R4),RMERRS ;SAVE THE FOUR REGISTERS THAT
941 044474 016437 000014 040514      MOV     RMER1(R4),RMERRS+2 ;WILL TELL US SOMETHING
942 044502 016437 000042 040516      MOV     RMER2(R4),RMERRS+4
943 044510 016437 000040 040520      MOV     RMMR2(R4),RMERRS+6
944 044516 004037 041072      JSR      RO,DRVINT ;INIT. THE STATE OF THE DRIVE
945 044522 000401      BR       1$      ;TAKE ERROR EXIT
946 044524 000207      RTS      PC       ;RETURN
947 044526 005726      1$: TST      (SP)+      ;POP PC OFF OF THE STACK
948 044530 000674      BR       SC8       ;PROCESS THE PARITY ERROR
949
950 044532 006301      SC13: ASL      R1       ;SETUP TO ADDRESS WORDS
951 044534 012761 177777 040614      MOV     #-1,TIMER(R1) ;STOP THE TIMER
952 044542 006201      ASR      R1
953 044544 010164 000010      MOV     R1,RMCS2(R4) ;SELECT THE DRIVE
954 044550 116164 040636 000016      MOV     ATABIT(R1),RMAS(R4) ;CLEAR THE ATTENTION BIT
955 044556 105761 040552      1$: TSTB     DPINT(R1)      ;INITIALIZING THE DRIVE ?

```

```

956 044562 001424      BEQ      2$      ;BR IF NOT
957 044564 105061 040552 CLR      DPINT(R1)      ;CLEAR THE INIT INDICATOR
958 044570 004037 041072 JSR      R0,DRVINT      ;GO INIT THE DRIVE
959 044574 000240      NOP      ;DUMMY PARITY ERROR RETURN
960 044576 105761 040532 TST      DRVSTA(R1)      ;DRIVE ONLINE ?
961 044602 003014      BGT      2$      ;BR IF YES -- START ORDER
962 044604 005702      TST      R2      ;QUEUE ENTRY FOR THE DRIVE
963 044606 001426      BEQ      3$      ;BR IF NOT
964 044610 004737 046510 JSR      PC,GETREQ      ;GET DPB ADDRESS
965 044614 052762 140000 000016 BIS      #BIT15,BIT14,16(R2) ;INFORM USER THAT DRIVE OFFLINE
966 044622 004737 045654 JSR      PC,SVH70      ;SAVE THE REGISTERS
967 044626 004737 046532 JSR      PC,POPQUE      ;REMOVE THE QUEUE
968 044632 000414      BR      3$
969 044634 032764 004000 000000 2$: BIF      #BIT11,RMCS1(R4)      ;DVA SET ?
970 044642 001006      BNE      4$      ;SET THEN CALL OPT
971 044644 006301      ASL      R1
972 044646 012761 035230 040614 MOV      #15000.,TIMER(R1) ;START 15. SECOND TIMER
973 044654 006201      ASR      R1
974 044656 000402      BR      3$
975 044660 004737 041700 4$: JSR      PC,OPT      ;START THE PENDING REQUEST
976 044664 000137 043574 3$: JMP      SC4      ;PROCESS OTHER DRIVES
977
978      ;RM TIMER ROUTINE
979      ;CALL
980      ;
981      ;
982      ;
983      ;
984      ;
985      ;
986 044670 005737 040576      RMTMR: TST      ACTDRV      ;CHECK 'ACTDRV & ACTSTR'
987 044674 001027      BNE      4$      ;IF NON ZERO EXIT
988 044676 112737 000001 040577 MOV      #1,ACTSTR      ;SET 'ACTSTR'
989 044704 104412      SAVREG      ;SAVE R0 - R5
990 044706 005001      CLR      R1      ;START WITH DRIVE 0
991 044710 005003      CLR      R3
992 044712 005763 040614 1$: TST      TIMER(R3)      ;IS THE TIMER RUNNING?
993 044716 002406      BLT      2$      ;BR IF NO
994 044720 166663 000002 040614 SUB      2(SP),TIMER(R3) ;COUNT THE INTERVAL
995 044726 003002      BGT      2$      ;BR IF NO SOFTWARE TIMEOUT
996 044730 004737 044760 JSR      PC,STO      ;CALL SOFTWARE TIMEOUT ROUTINE
997 044734 005201 2$: INC      R1      ;MOVE TO NEXT DRIVE
998 044736 005723      TST      (R3)+
999 044740 022701 000010      CMP      #8.,R1      ;OUT OF DRIVES?
1000 044744 003362      BGT      1$      ;BR IF NO
1001 044746 104413 3$: RESREG      ;RESTORE R0 - R5
1002 044750 105037 040577      CLR      ACTSTR      ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
1003 044754 012616 4$: MOV      (SP)+,(SP) ;ADJUST THE STACK
1004 044756 000207      RTS      PC      ;RETURN
1005
1006      ;SOFTWARE TIMEOUT ROUTINE
1007      ;
1008      ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
1009      ;OR GREATER
1010      ;
1011      ;CALL:
1012      ;
1013      ;
1014      ;
1015      ;
1016      ;
1017      ;
1018      ;
1019      ;
1020      ;
1021      ;
1022      ;
1023      ;
1024      ;
1025      ;
1026      ;
1027      ;
1028      ;
1029      ;
1030      ;
1031      ;
1032      ;
1033      ;
1034      ;
1035      ;
1036      ;
1037      ;
1038      ;
1039      ;
1040      ;
1041      ;
1042      ;
1043      ;
1044      ;
1045      ;
1046      ;
1047      ;
1048      ;
1049      ;
1050      ;
1051      ;
1052      ;
1053      ;
1054      ;
1055      ;
1056      ;
1057      ;
1058      ;
1059      ;
1060      ;
1061      ;
1062      ;
1063      ;
1064      ;
1065      ;
1066      ;
1067      ;
1068      ;
1069      ;
1070      ;
1071      ;
1072      ;
1073      ;
1074      ;
1075      ;
1076      ;
1077      ;
1078      ;
1079      ;
1080      ;
1081      ;
1082      ;
1083      ;
1084      ;
1085      ;
1086      ;
1087      ;
1088      ;
1089      ;
1090      ;
1091      ;
1092      ;
1093      ;
1094      ;
1095      ;
1096      ;
1097      ;
1098      ;
1099      ;
1100      ;
1101      ;
1102      ;
1103      ;
1104      ;
1105      ;
1106      ;
1107      ;
1108      ;
1109      ;
1110      ;
1111      ;
1112      ;
1113      ;
1114      ;
1115      ;
1116      ;
1117      ;
1118      ;
1119      ;
1120      ;
1121      ;
1122      ;
1123      ;
1124      ;
1125      ;
1126      ;
1127      ;
1128      ;
1129      ;
1130      ;
1131      ;
1132      ;
1133      ;
1134      ;
1135      ;
1136      ;
1137      ;
1138      ;
1139      ;
1140      ;
1141      ;
1142      ;
1143      ;
1144      ;
1145      ;
1146      ;
1147      ;
1148      ;
1149      ;
1150      ;
1151      ;
1152      ;
1153      ;
1154      ;
1155      ;
1156      ;
1157      ;
1158      ;
1159      ;
1160      ;
1161      ;
1162      ;
1163      ;
1164      ;
1165      ;
1166      ;
1167      ;
1168      ;
1169      ;
1170      ;
1171      ;
1172      ;
1173      ;
1174      ;
1175      ;
1176      ;
1177      ;
1178      ;
1179      ;
1180      ;
1181      ;
1182      ;
1183      ;
1184      ;
1185      ;
1186      ;
1187      ;
1188      ;
1189      ;
1190      ;
1191      ;
1192      ;
1193      ;
1194      ;
1195      ;
1196      ;
1197      ;
1198      ;
1199      ;
1200      ;
1201      ;
1202      ;
1203      ;
1204      ;
1205      ;
1206      ;
1207      ;
1208      ;
1209      ;
1210      ;
1211      ;
1212      ;
1213      ;
1214      ;
1215      ;
1216      ;
1217      ;
1218      ;
1219      ;
1220      ;
1221      ;
1222      ;
1223      ;
1224      ;
1225      ;
1226      ;
1227      ;
1228      ;
1229      ;
1230      ;
1231      ;
1232      ;
1233      ;
1234      ;
1235      ;
1236      ;
1237      ;
1238      ;
1239      ;
1240      ;
1241      ;
1242      ;
1243      ;
1244      ;
1245      ;
1246      ;
1247      ;
1248      ;
1249      ;
1250      ;
1251      ;
1252      ;
1253      ;
1254      ;
1255      ;
1256      ;
1257      ;
1258      ;
1259      ;
1260      ;
1261      ;
1262      ;
1263      ;
1264      ;
1265      ;
1266      ;
1267      ;
1268      ;
1269      ;
1270      ;
1271      ;
1272      ;
1273      ;
1274      ;
1275      ;
1276      ;
1277      ;
1278      ;
1279      ;
1280      ;
1281      ;
1282      ;
1283      ;
1284      ;
1285      ;
1286      ;
1287      ;
1288      ;
1289      ;
1290      ;
1291      ;
1292      ;
1293      ;
1294      ;
1295      ;
1296      ;
1297      ;
1298      ;
1299      ;
1300      ;
1301      ;
1302      ;
1303      ;
1304      ;
1305      ;
1306      ;
1307      ;
1308      ;
1309      ;
1310      ;
1311      ;
1312      ;
1313      ;
1314      ;
1315      ;
1316      ;
1317      ;
1318      ;
1319      ;
1320      ;
1321      ;
1322      ;
1323      ;
1324      ;
1325      ;
1326      ;
1327      ;
1328      ;
1329      ;
1330      ;
1331      ;
1332      ;
1333      ;
1334      ;
1335      ;
1336      ;
1337      ;
1338      ;
1339      ;
1340      ;
1341      ;
1342      ;
1343      ;
1344      ;
1345      ;
1346      ;
1347      ;
1348      ;
1349      ;
1350      ;
1351      ;
1352      ;
1353      ;
1354      ;
1355      ;
1356      ;
1357      ;
1358      ;
1359      ;
1360      ;
1361      ;
1362      ;
1363      ;
1364      ;
1365      ;
1366      ;
1367      ;
1368      ;
1369      ;
1370      ;
1371      ;
1372      ;
1373      ;
1374      ;
1375      ;
1376      ;
1377      ;
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      ;
1434      ;
1435      ;
1436      ;
1437      ;
1438      ;
1439      ;
1440      ;
1441      ;
1442      ;
1443      ;
1444      ;
1445      ;
1446      ;
1447      ;
1448      ;
1449      ;
1450      ;
1451      ;
1452      ;
1453      ;
1454      ;
1455      ;
1456      ;
1457      ;
1458      ;
1459      ;
1460      ;
1461      ;
1462      ;
1463      ;
1464      ;
1465      ;
1466      ;
1467      ;
1468      ;
1469      ;
1470      ;
1471      ;
1472      ;
1473      ;
1474      ;
1475      ;
1476      ;
1477      ;
1478      ;
1479      ;
1480      ;
1481      ;
1482      ;
1483      ;
1484      ;
1485      ;
1486      ;
1487      ;
1488      ;
1489      ;
1490      ;
1491      ;
1492      ;
1493      ;
1494      ;
1495      ;
1496      ;
1497      ;
1498      ;
1499      ;
1500      ;
1501      ;
1502      ;
1503      ;
1504      ;
1505      ;
1506      ;
1507      ;
1508      ;
1509      ;
1510      ;
1511      ;
1512      ;
1513      ;
1514      ;
1515      ;
1516      ;
1517      ;
1518      ;
1519      ;
1520      ;
1521      ;
1522      ;
1523      ;
1524      ;
1525      ;
1526      ;
1527      ;
1528      ;
1529      ;
1530      ;
1531      ;
1532      ;
1533      ;
1534      ;
1535      ;
1536      ;
1537      ;
1538      ;
1539      ;
1540      ;
1541      ;
1542      ;
1543      ;
1544      ;
1545      ;
1546      ;
1547      ;
1548      ;
1549      ;
1550      ;
1551      ;
1552      ;
1553      ;
1554      ;
1555      ;
1556      ;
1557      ;
1558      ;
1559      ;
1560      ;
1561      ;
1562      ;
1563      ;
1564      ;
1565      ;
1566      ;
1567      ;
1568      ;
1569      ;
1570      ;
1571      ;
1572      ;
1573      ;
1574      ;
1575      ;
1576      ;
1577      ;
1578      ;
1579      ;
1580      ;
1581      ;
1582      ;
1583      ;
1584      ;
1585      ;
1586      ;
1587      ;
1588      ;
1589      ;
1590      ;
1591      ;
1592      ;
1593      ;
1594      ;
1595      ;
1596      ;
1597      ;
1598      ;
1599      ;
1600      ;
1601      ;
1602      ;
1603      ;
1604      ;
1605      ;
1606      ;
1607      ;
1608      ;
1609      ;
1610      ;
1611      ;
1612      ;
1613      ;
1614      ;
1615      ;
1616      ;
1617      ;
1618      ;
1619      ;
1620      ;
1621      ;
1622      ;
1623      ;
1624      ;
1625      ;
1626      ;
1627      ;
1628      ;
1629      ;
1630      ;
1631      ;
1632      ;
1633      ;
1634      ;
1635      ;
1636      ;
1637      ;
1638      ;
1639      ;
1640      ;
1641      ;
1642      ;
1643      ;
1644      ;
1645      ;
1646      ;
1647      ;
1648      ;
1649      ;
1650      ;
1651      ;
1652      ;
1653      ;
1654      ;
1655      ;
1656      ;
1657      ;
1658      ;
1659      ;
1660      ;
1661      ;
1662      ;
1663      ;
1664      ;
1665      ;
1666      ;
1667      ;
1668      ;
1669      ;
1670      ;
1671      ;
1672      ;
1673      ;
1674      ;
1675      ;
1676      ;
1677      ;
1678      ;
1679      ;
1680      ;
1681      ;
1682      ;
1683      ;
1684      ;
1685      ;
1686      ;
1687      ;
1688      ;
1689      ;
1690      ;
1691      ;
1692      ;
1693      ;
1694      ;
1695      ;
1696      ;
1697      ;
1698      ;
1699      ;
1700      ;
1701      ;
1702      ;
1703      ;
1704      ;
1705      ;
1706      ;
1707      ;
1708      ;
1709      ;
1710      ;
1711      ;
1712      ;
1713      ;
1714      ;
1715      ;
1716      ;
1717      ;
1718      ;
1719      ;
1720      ;
1721      ;
1722      ;
1723      ;
1724      ;
1725      ;
1726      ;
1727      ;
1728      ;
1729      ;
1730      ;
1731      ;
1732      ;
1733      ;
1734      ;
1735      ;
1736      ;
1737      ;
1738      ;
1739      ;
1740      ;
1741      ;
1742      ;
1743      ;
1744      ;
1745      ;
1746      ;
1747      ;
1748      ;
1749      ;
1750      ;
1751      ;
1752      ;
1753      ;
1754      ;
1755      ;
1756      ;
1757      ;
1758      ;
1759      ;
1760      ;
1761      ;
1762      ;
1763      ;
1764      ;
1765      ;
1766      ;
1767      ;
1768      ;
1769      ;
1770      ;
1771      ;
1772      ;
1773      ;
1774      ;
1775      ;
1776      ;
1777      ;
1778      ;
1779      ;
1780      ;
1781      ;
1782      ;
1783      ;
1784      ;
1785      ;
1786      ;
1787      ;
1788      ;
1789      ;
1790      ;
1791      ;
1792      ;
1793      ;
1794      ;
1795      ;
1796      ;
1797      ;
1798      ;
1799      ;
1800      ;
1801      ;
1802      ;
1803      ;
1804      ;
1805      ;
1806      ;
1807      ;
1808      ;
1809      ;
1810      ;
1811      ;
1812      ;
1813      ;
1814      ;
1815      ;
1816      ;
1817      ;
1818      ;
1819      ;
1820      ;
1821      ;
1822      ;
1823      ;
1824      ;
1825      ;
1826      ;
1827      ;
1828      ;
1829      ;
1830      ;
1831      ;
1832      ;
1833      ;
1834      ;
1835      ;
1836      ;
1837      ;
1838      ;
1839      ;
1840      ;
1841      ;
1842      ;
1843      ;
1844      ;
1845      ;
1846      ;
1847      ;
1848      ;
1849      ;
1850      ;
1851      ;
1852      ;
1853      ;
1854      ;
1855      ;
1856      ;
1857      ;
1858      ;
1859      ;
1860      ;
1861      ;
1862      ;
1863      ;
1864      ;
1865      ;
1866      ;
1867      ;
1868      ;
1869      ;
1870      ;
1871      ;
1872      ;
1873      ;
1874      ;
1875      ;
1876      ;
1877      ;
1878      ;
1879      ;
1880      ;
1881      ;
1882      ;
1883      ;
1884      ;
1885      ;
1886      ;
1887      ;
1888      ;
1889      ;
1890      ;
1891      ;
1892      ;
1893      ;
1894      ;
1895      ;
1896      ;
1897      ;
1898      ;
1899      ;
1900      ;
1901      ;
1902      ;
1903      ;
1904      ;
1905      ;
1906      ;
1907      ;
1908      ;
1909      ;
1910      ;
1911      ;
1912      ;
1913      ;
1914      ;
1915      ;
1916      ;
1917      ;
1918      ;
1919      ;
1920      ;
1921      ;
1922      ;
1923      ;
1924      ;
1925      ;
1926      ;
1927      ;
1928      ;
1929      ;
1930      ;
1931      ;
1932      ;
1933      ;
1934      ;
1935      ;
1936      ;
1937      ;
1938      ;
1939      ;
1940      ;
1941      ;
1942      ;
1943      ;
1944      ;
1945      ;
1946      ;
1947      ;
1948      ;
1949      ;
1950      ;
1951      ;
1952      ;
1953      ;
1954      ;
1955      ;
1956      ;
1957      ;
1958      ;
1959      ;
1960      ;
1961      ;
1962      ;
1963      ;
1964      ;
1965      ;
1966      ;
1967      ;
1968      ;
1969      ;
1970      ;
1971      ;
1972      ;
1973      ;
1974      ;
1975      ;
1976      ;
1977      ;
1978      ;
1979      ;
1980      ;
1981      ;
1982      ;
1983      ;
1984      ;
1985      ;
1986      ;
1987      ;
1988      ;
1989      ;
1990      ;
1991      ;
1992      ;
1993      ;
1994      ;
1995      ;
1996      ;
1997      ;
1998      ;
1999      ;
2000      ;
2001      ;
2002      ;
2003      ;
2004      ;
2005      ;
2006      ;
2007      ;
2008      ;
2009      ;
2010      ;
2011      ;
2012      ;
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      ;
2069      ;
2070      ;
2071      ;
2072      ;
2073      ;
2074      ;
2075      ;
2076      ;
2077      ;
2078      ;
2079      ;
2080      ;
2081      ;
2082      ;
2083      ;
2084      ;
2085      ;
2086      ;
2087      ;
2088      ;
2089      ;
2090      ;
2091      ;
2092      ;
2093      ;
2094      ;
2095      ;
2096      ;
2097      ;
2098      ;
2099      ;
2100      ;
2101      ;
2102      ;
2103      ;
2104      ;
2105      ;
2106      ;
2107      ;
2108      ;
2109      ;
2110      ;
2111      ;
2112      ;
2113      ;
2114      ;
2115      ;
2116      ;
2117      ;
2118      ;
2119      ;
2120      ;
2121      ;
2122      ;
2123      ;
2124      ;
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      ;
2181      ;
2182      ;
2183      ;
2184      ;
2185      ;
2186      ;
2187      ;
2188      ;
2189      ;
2190      ;
2191      ;
2192      ;
2193      ;
2194      ;
2195      ;
2196      ;
2197      ;
2198      ;
2199      ;
2200      ;
2201      ;
2202      ;
2203      ;
2204      ;
2205      ;
2206      ;
2207      ;
2208      ;
2209      ;
2210      ;
2211      ;
2212      ;
2213      ;
2214      ;
2215      ;
2216      ;
2217      ;
2218      ;
2219      ;
2220      ;
2221      ;
2222      ;
2223      ;
2224      ;
2225      ;
2226      ;
2227      ;
2228      ;
2229      ;
2230      ;
2231      ;
2232      ;
2233      ;
2234      ;
2235      ;
2236      ;
2237      ;
2238      ;
2239      ;
2240      ;
2241      ;
2242      ;
2243      ;
2244      ;
2245      ;
2246      ;
2247      ;
2248      ;
2249      ;
2250      ;
2251      ;
2252      ;
2253      ;
2254      ;
2255      ;
2256      ;
2257      ;
2258      ;
2259      ;
2260      ;
2261      ;
2262      ;
2263      ;
2264      ;
2265      ;
2266      ;
2267      ;
2268      ;
2269      ;
2270      ;
2271      ;
2272      ;
2273      ;
2274      ;
2275      ;
2276      ;
2277      ;
2278      ;
2279      ;
2280      ;
2281      ;
2282      ;
2283      ;
2284      ;
2285      ;
2286      ;
2287      ;
2288      ;
2289      ;
2290      ;
2291      ;
2292      ;
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      ;
2349      ;
2350      ;
2351      ;
2352      ;
2353      ;
2354      ;
2355      ;
2356      ;
2357      ;
2358      ;
2359      ;
2360      ;
2361      ;
2362      ;
2363      ;
2364      ;
2365      ;
2366      ;
2367      ;
2368      ;
2369      ;
2370      ;
2371      ;
2372      ;
2373      ;
2374      ;
2375      ;
2376      ;
2377      ;
2378      ;
2379      ;
2380      ;
2381      ;
2382      ;
2383      ;
2384      ;
2385      ;
2386      ;
2387      ;
2388      ;
2389      ;
2390      ;
2391      ;
2392      ;
2393      ;
2394      ;
2395      ;
2396      ;
2397      ;
2398      ;
2399      ;
2400      ;
2401      ;
2402      ;
2403      ;
2404      ;
2405      ;
2406      ;
2407      ;
2408      ;
2409      ;
2410      ;
2411      ;
2412      ;
2413      ;
2414      ;
2415      ;
2416      ;
2417      ;
2418      ;
2419      ;
2420      ;
2421      ;
2422      ;
2423      ;
2424      ;
2425      ;
2426      ;
2427      ;
2428      ;
2429      ;
2430      ;
2431      ;
2432      ;
2433      ;
2434      ;
2435      ;
2436      ;
2437      ;
2438      ;
2439      ;
2440      ;
2441      ;
2442      ;
2443      ;
2444      ;
2445      ;
2446      ;
2447      ;
2448      ;
2449      ;
2450      ;
2451      ;
2452      ;
2453      ;
2454      ;
2455      ;
2456      ;
2457      ;
2458      ;
2459      ;
2460      ;
2461      ;
2462      ;
2463      ;
2464      ;
2465      ;
2466      ;
2467      ;
2468      ;
2469      ;
2470      ;
2471      ;
2472      ;
2473      ;
2474      ;
2475      ;
2476      ;
2477      ;
2478      ;
2479      ;
2480      ;
2481      ;
2482      ;
2483      ;
2484      ;
2485      ;
2486      ;
2487      ;
2488      ;
2489      ;
2490      ;
2491      ;
2492      ;
2493      ;
2494      ;
2495      ;
2496      ;
2497      ;
2498      ;
2499      ;
2500      ;
2501      ;
2502      ;
2503      ;
2504      ;
2505      ;
2506      ;
2507      ;
2508      ;
2509      ;
2510      ;
2511      ;
2512      ;
2513      ;
2514      ;
2515      ;
2516      ;
2517      ;
2518      ;
2519      ;
2520      ;
2521      ;
2522      ;
2523      ;
2524      ;
2525      ;
2526      ;
2527      ;
2528      ;
2529      ;
2530      ;
2531      ;
2532      ;
2533      ;
2534     
```

```

1016 044760 010146          STO:  MOV    R1,-(SP)      ;SAVE R1
1017 044762 010246          MOV    R2,-(SP)      ;SAVE R2
1018 044764 010346          MOV    R3,-(SP)      ;SAVE R3
1019 044766 010446          MOV    R4,-(SP)      ;SAVE R4
1020 044770 013704 040650    MOV    RMADR,R4      ;GET ADDRESS OF 'RMCS1'
1021 044774 010164 000010    MOV    R1,RMCS2(R4)  ;SELECT THE DRIVE
1022 045000 004037 045302    JSR    R0,RD.RM      ;READ 'DRIVE STATUS REG'
1023 045004 000012          RMD5
1027 045006 045270          ST09
1028 045010 105726          TSTB    (SP)+        ;IS 'DRY'=1?
1029 045012 100436          BMI     ST02          ;BR IF YES
1030 045014 105761 040552    ST01:  TSTB    DPINT(R1)  ;TRYING TO INITIALIZE THE DRIVE ?
1031 045020 001033          BNE     ST02          ;BR IF YES
1032 045022 105761 040562    TSTB    DPRQS(R1)  ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
1033 045026 001030          BNE     ST02          ;BR IF YES
1034 045030 013702 040572    MOV     TRNSWT,R2      ;PICKUP TRANSFER WAIT QUEUE
1035 045034 020137 040634    CMP     R1,DTUW      ;TRANSFER UNDERWAY ON THIS DRIVE?
1036 045040 001404          BEQ     1$          ;BR IF YES
1037 045042 000137 045270    JMP     ST09          ;IF NOT DON'T BOTHER DRIVES
1038 045046 004737 046510    JSR     PC,GETREQ    ;GET DPB ADDRESS
1039 045052 052762 101000 000016 1$:  BIS     #BIT15:BIT09,16(R2) ;SET THE ERROR FLAGS
1040 045060 004737 045654    JSR     PC,SVRH70    ;SAVE RH/RM REGISTERS
1044 045064 105061 040522    CLRB    DRVACT(R1)  ;DRIVE IS IDLE
1045 045070 105061 040600    CLRB    ULDFLG(R1)  ;CLEAR THE UNLOAD FLAG
1046 045074 005037 040572    CLR     TRNSWT      ;CLEAR DPB ADDRESS
1047 045100 012737 177777 040634    MOV     #-1,DTUW      ;CLEAR THE TRANSFER DRIVE #
1048 045106 000470          BR      ST09          ;DON'T BOTHER OTHER DRIVES
1049
1050 045110 116405 000016    ST02:  MOVB   RMAS(R4),R5  ;READ ATTENTION REG
1051 045114 136105 040636    BITB    ATABIT(R1),R5 ;IS ATTENTION FOR THIS DRIVE UP ?
1052 045120 001007          BNE     ST03          ;YES
1053 045122 105761 040552    TSTB    DPINT(R1)  ;TRYING TO INITIALIZE THE DRIVE ?
1054 045126 001021          BNE     ST06          ;BR IF YES - DRIVE NOT ONLINE
1055 045130 105761 040562    TSTB    DPRQS(R1)  ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
1056 045134 001035          BNE     ST07          ;BR IF YES - NO RESPONSE TO REQUEST
1057 045136 000454          BR      ST09          ;OTHER WISE EXIT
1058
1059 045140 105761 040552    ST03:  TSTB    DPINT(R1)  ;INITIALIZING THE DRIVE ?
1060 045144 001003          BNE     1$          ;BR IF INIT PENDING
1061 045146 105761 040562    TSTB    DPRQS(R1)  ;PORT REQUEST PENDING ?
1062 045152 001446          BEQ     ST09          ;BR IF NOT
1063 045154 012763 177777 040614 1$:  MOV     #-1,TIMER(R3) ;STOP THE TIMER
1064 045162 000442          BR      ST09          ;EXIT
1065
1066 045164 004737 043056    ST05:  JSR     PC,C18      ;GO HANDLE THE PARITY ERROR
1067 045170 000437          BR      ST09
1068
1069 045172 105061 040552    ST06:  CLRB    DPINT(R1)  ;CLEAR THE INITIALIZE INDICATOR
1070 045176 105061 040532    CLRB    DRVSTA(R1) ;SET UNIT OFFLINE
1071 045202 012763 177777 040614    MOV     #-1,TIMER(R3) ;STOP THE TIMER
1072 045210 004737 046510    JSR     PC,GETREQ    ;GET THE DPB ADDRESS
1073 045214 005702          TST     R2          ;REQUEST IN QUEUE ?
1074 045216 001424          BEQ     ST09          ;BR IF NOT
1075 045220 052762 140000 000016    BIS     #BIT15:BIT14,16(R2) ;INFORM THE USER DRIVE NOT AVAILABLE
1076 045226 000414          BR      ST08          ;FINISH
1077
1078 045230 012763 177777 040614 ST07:  MOV     #-1,TIMER(R3) ;STOP THE TIMER

```

```

1079 045236 105061 040562 CLR8 DPRQS(R1) ;CLEAR PORT REQUEST INDICATOR
1080 045242 004737 046510 JSR PC,GETREQ ;GET DPB ADDRESS
1081 045246 005702 TST R2 ;QUEUE ENTRY FOR DRIVE ?
1082 045250 001407 BEQ ST09 ;BR IF NONE
1083 045252 012762 100004 000016 MOV #BIT15:BIT2,16(R2) ;INFORM USER OF PORT REQUEST ERROR
1084 045260 004737 046414 ST08: JSR PC,EMPTYQ ;CLEAR THE QUEUE FOR THE DRIVE
1085 045264 004737 045654 JSR PC,SVRH70 ;SAVE THE REGISTERS
1086 045270 012604 ST09: MOV (SP)+,R4 ;RESTORE R4
1087 045272 012603 MOV (SP)+,R3 ;RESTORE R3
1088 045274 012602 MOV (SP)+,R2 ;RESTORE R2
1089 045276 012601 MOV (SP)+,R1 ;RESTORE R1
1090 045300 000207 RTS PC ;RETURN
1091
1092 ;ROUTINE TO READ A RH/RM REGISTER
1093
1094 :CALL
1095 JSR R0,RD.RM ;GO READ A REGISTER
1096 INDEX ;REG. INDEX FROM BASE
1097 ERRADR ;ERROR ADDRESS--PROCESS ERROR STARTING
1098 ;AT THIS ADDRESS
1099 RETURN ;CONTENTS OF REG. IS ON THE STACK
1100
1101 045302 013737 040646 045450 RD.RM: MOV MCPEMX,RD.RM2 ;MAX. RETRYS ALLOWED
1102 045310 011646 MOV (SP),-(SP) ;SAVE R0 FOR RETURN
1103 045312 013737 040650 045326 MOV RMADR,RD.ADR ;FORM THE DESIRED ADDRESS
1104 045320 062037 045326 ADD (R0)+,RD.ADR ;USING THE BASE AND THE INDEX
1105 045324 013727 RD.RM1: MOV @ (PC)+,(PC)+ ;READ THE DESIRED REGISTER OF THE RM DRIVE
1106 045326 000000 RD.ADR: .WORD 0 ;ADDRESS IS FORMED HERE
1107 045330 000000 RD.WRD: .WORD 0 ;REG. CONTENTS PUT HERE
1108 045332 013766 045330 000002 MOV RD.WRD,2(SP) ;RETURN IT TO THE USER
1109 045340 013746 040650 MOV RMADR,-(SP) ;PUT THE ADDRESS ON THE STACK
1110 045344 062716 000010 ADD #RMCS2,(SP) ;FORM THE ADDRESS OF RMCS2
1111 045350 032736 010000 BIT #BIT12,@(SP)+ ;CHECK THE 'NED' BIT
1112 045354 001037 BNE RD.RM3 ;BR IF DRIVE NON-EXISTENT
1113 045356 017746 173266 MOV @RMADR,-(SP) ;READ RMCS1
1114 045362 032716 020000 BIT #BIT13,(SP) ;DID MCPE SET?
1115 045366 001002 BNE 1$ ;BR IF YES
1116 045370 022620 CMP (SP)+,(R0)+ ;ADJUST FOR RETURN
1117 045372 000432 BR RD.RM4 ;EXIT
1118 045374
1119 045374 004037 046600 1$: JSR R0,ES.SAV ;SAVE THE ADDRESS IN '$ESCAPE'
1120 045400 104003 EMT 3 ;REPORT 'MCPE' ERROR
1121 045402 005737 040634 TST DTUW ;DATA TRANSFER UNDERWAY?
1122 045406 100405 BMI 2$ ;NO
1123 045410 032716 040000 BIT #BIT14,(SP) ;'TRE' = 1 ?
1124 045414 001402 BEQ 2$ ;NO
1125 045416 005726 TST (SP)+ ;YES--CLEAN OFF THE STACK AND
1126 045420 000415 BR RD.RM3 ;TAKE THE FATAL ERROR EXIT
1127 045422 052716 040000 2$: BIS #BIT14,(SP) ;CLEAR 'MCPE' BY SENDING A '1' TO 'TRE'
1128 045426 000316 SWAB (SP) ;POSITION BEFORE WRITING
1129 045430 013737 040650 045444 MOV RMADR,3$ ;FORM ADDRESS OF HIGH BYTE
1130 045436 005237 045444 INC 3$
1131 045442 112637 MOV (SP)+,@(PC)+ ;WRITE THE HIGH BYTE OF RMCS1
1132 045444 000000 3$: .WORD 0 ;ADDRESS STORAGE
1133 045446 005327 DEC (PC)+ ;EXCEEDED MAX. RETRYS
1134 045450 000003 PD.RM2: .WORD 3
1135 045452 002324 BGE RD.RM1 ;BR IF NO

```

```

1134 045454 011000 RD.RM3: MOV (R0),R0 ;FATAL ERROR EXIT
1135 045456 012616 MOV (SP)+,(SP)
1136 045460 000200 RD.RM4: RTS R0
1137
1138 ;ROUTINE TO WRITE A REGISTER
1139 ;CALL
1140
1141 MOV DATA,-(SP) ;DATA TO BE LOADED ON THE STACK
1142 JSR R0,WRT.RM ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
1143 INDEX ;INDEX OF THE REGISTER TO BE LOADED
1144 ERRADR ;ADDRESS TO RETURN TO ON AN ERROR
1145 RETURN ;ERROR FREE RETURN
1146
1147 045462 013737 040646 045640 WRT.RM: MOV MCPMX,WRT.R2 ;MAX RETRYS ALLOWED
1148 045470 016637 000002 045550 MOV 2(SP),WRT.WD ;SAVE THE WORD TO WRITE
1149 045476 012616 MOV (SP)+,(SP) ;ADJUST THE STACK
1150 045500 012037 045552 MOV (R0)+,WRT.AD ;GET INDEX OF REGISTER TO BE WRITTEN
1151 045504 001015 BNE 1$ ;BR IF NOT RMCS1
1152 045506 122737 000150 045550 CMPB #150,WRT.WD ;IS THE COMMAND FOR DATA TRANSFERS?
1153 045514 002411 BLT 1$ ;YES--DON'T GET THE OLD A16 & A17, & PSEL
1154 045516 004037 045302 JSR R0,RD.RM ;NO---COMBINE A16&A17, & PSEL WITH
1155 045522 000000 RMCS1 ;THE COMMAND BEFORE SENDING IT TO
1156 045524 045644 WRT.R3 ;THE RH/RM
1157 045526 000316 SWAB (SP)
1158 045530 042716 177770 BIC #^C7,(SP)
1159 045534 112637 045551 MOVB (SP)+,WRT.WD+1
1160 045540 063737 040650 045552 1$: ADD RMADR,WRT.AD ;FORM THE ADDRESS OF THE DISK REG.
1161 045546 012737 WRT.R1: MOV (PC)+,@(PC)+ ;LOAD THE DESIRED REG.
1162 045550 000000 WRT.WD: .WORD 0 ;WORD TO WRITE GOES HERE
1163 045552 000000 WRT.AD: .WORD 0 ;ADDRESS IS FORMED HERE
1164 045554 013746 040650 MOV RMADR,-(SP) ;PUT THE ADDRESS ON THE STACK
1165 045560 062716 000010 ADD #RMCS2,(SP) ;FORM THE ADDRESS OF RMCS2
1166 045564 032736 010000 BIT #BIT12,@(SP)+ ;CHECK THE 'NED' BIT
1167 045570 001025 BNE WRT.R3 ;BR IF DRIVE NON-EXISTENT
1168 045572 004037 045302 JSR R0,RD.RM ;CHECK FOR PARITY ERROR ON WRITE
1169 045576 000014 RMER1
1170 045600 045644 WRT.R3
1171 045602 032726 000010 BIT #BIT03,(SP)+
1172 045606 001420 BEQ WRT.R4 ;BR IF 'PAR=0'
1173 045610 016037 177776 045622 MOV -2(R0),1$ ;PICKUP THE INDEX
1174 045616 004037 045302 JSR R0,RD.RM ;READ THE REG.
1175 045622 000000 1$: .WORD 0 ;REG. INDEX
1176 045624 045644 WRT.R3 ;RETURN TO THIS ADDRESS ON ERROR
1177 045626 004037 046600 JSR R0,ES.SAV ;SAVE THE ADDRESS IN '$ESCAPE'
1178 045632 104004 EMT 4 ;REPORT THE PARITY ON WRITE ERROR
1179 045636 005726 TST (SP)+ ;CLEAR OFF THE STACK
1180 045640 000003 WRT.R2: .WORD 3 ;DECREMENT THE ERROR COUNT
1181 045642 002341 BGE WRT.R1 ;RETRY COUNTER
1182 045644 011000 WRT.R3: MOV (R0),R0 ;TRY AGAIN IF NOT FINISHED
1183 045646 000401 BR WRT.R5 ;TAKE THE 'PARITY ON WRITE' ERROR EXIT
1184 045650 005720 WRT.R4: TST (R0)+ ;EXIT
1185 045652 000200 WRT.R5: RTS R0 ;ADJUST FOR ERROR FREE EXIT
1186
1187 ;ROUTINE TO SAVE THE RH/RM REGISTERS AS PER DPB+14
1188 ;CALL
1189

```

```

1190      :      MOV      #DPBNUM,R2      ;DPB POINTER TO R2
1191      :      JSR      PC,SVRH70      ;SAVE THE DRIVES REG'S
1192
1193 045654 104412      SVRH70: SAVREG      ;SAVE R0 - R5
1194 045656 005702      TST      R2      ;QUEUE ENTRY FOR THE DRIVE ?
1195 045660 001442      BEQ      6$      ;BR IF NONE
1196 045662 013704 040650      MOV      RMADR,R4
1197 045666 111264 000010      MOV      (R2),RMCS2(R4) ;SELECT DRIVE
1198 045672 016203 000014      MOV      14(R2),R3      ;GET THE ERROR TABLE POINTER
1199 045676 001433      BEQ      6$      ;EXIT IF NO ADDRESS
1200 045700 005037 045734      CLR      3$      ;COUNTER & POINTER
1201 045704 023727 045734 000022 1$:      CMP      3$,#RMD8      ;REACHED THE BUFFER REGISTER ?
1202 045712 001006      BNE      2$      ;BR IF NOT
1203 045714 032764 000200 000010      BIT      #BIT07,RMCS2(R4) ;'OR' SET ?
1204 045722 001002      BNE      2$      ;BR IF SET
1205 045724 005023      CLR      (R3)+      ;STORE RMD8 AS ZEROES
1206 045726 000405      BR      4$      ;CONTINUE
1207 045730 004037 045302      2$:      JSR      R0,RD.RM      ;READ THE SELECTED REGISTER
1208 045734 000000      3$:      .WORD      0      ;REGISTER INDEX
1209 045736 045762      5$:      MOV      (SP)+,(R3)+      ;ERROR RETURN ADDRESS
1210 045740 012623      MOV      (SP)+,(R3)+      ;STORE THE REGISTER CONTENTS
1211 045742 023727 045734 000046 4$:      CMP      3$,#RMEC2      ;REACHED THE END ?
1212 045750 001406      BEQ      6$      ;BR IF YES
1213 045752 062737 000002 045734      ADD      #2,3$      ;INCREMENT THE REGISTER INDEX
1214 045760 000751      BR      1$      ;CONTINUE READING THE REGISTERS
1215 045762 004737 042756      5$:      JSR      PC,C17      ;PROCESS THE UNCORRECTABLE PARITY ERROR
1226 045766 104413      6$:      RESREG      ;RESTORE R0 - R5
1228 045770 000207      RTS      PC      ;RETURN
1229
1230      ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A 'TRE'
1231      ;CALL
1232      :      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
1233      :      JSR      PC,SET.IE      ;SET 'IE'
1234      :      RETURN
1235
1236 045772 010446      SET.IE: MOV      R4,-(SP)      ;SAVE R4
1237 045774 013704 040650      MOV      RMADR,R4      ;PICKUP ADDRESS OF RMCS1
1238 046000 010164 000010      MOV      R1,RMCS2(R4)      ;SELECT DRIVE
1239 046004 011446      MOV      (R4),-(SP)      ;READ RMCS1
1240 046006 052716 040000      BIS      #BIT14,(SP)      ;SET THE 'TRE' BIT OF THE WORD READ
1241 046012 000316      SWAB      (SP)      ;ADJUST FOR DATO
1242 046014 112714 000100      MOV      #BIT06,(R4)      ;SET 'IE'
1243 046020 032764 010000 000010      BIT      #BIT12,RMCS2(R4) ;IS 'NED'=1?
1244 046026 001002      BNE      1$      ;YES--CLEAR 'TRE'
1245 046030 005726      TST      (SP)+      ;CLEAN OFF THE STACK
1246 046032 000402      BR      2$
1247 046034 112664 000001      1$:      MOV      (SP)+,1(R4)      ;CLEAR 'TRE'
1248 046040 012604      2$:      MOV      (SP)+,R4      ;RESTORE R4
1249 046042 000207      RTS      PC      ;RETURN TO CALLER
1250
1251      ;QUEUE COUNT
1252
1253 046044 000      QCNT: .BYTE      0      ;DRIVE 0
1254 046045 000      .BYTE      0      ;DRIVE 1
1255 046046 000      .BYTE      0      ;DRIVE 2
1256 046047 000      .BYTE      0      ;DRIVE 3
1257 046050 000      .BYTE      0      ;DRIVE 4

```

```

046051 000 .BYTE 0 ;DRIVE 5
046052 000 .BYTE 0 ;DRIVE 6
046053 000 .BYTE 0 ;DRIVE 7

1257
1258 ;QUEUE INPUT POINTERS
1259
1260 046054 046136 QINPT: .WORD QDRV0 ;DRIVE 0
1263 046056 046156 .WORD QDRV1 ;DRIVE 1
046060 046176 .WORD QDRV2 ;DRIVE 2
046062 046216 .WORD QDRV3 ;DRIVE 3
046064 046236 .WORD QDRV4 ;DRIVE 4
046066 046256 .WORD QDRV5 ;DRIVE 5
046070 046276 .WORD QDRV6 ;DRIVE 6
046072 046316 .WORD QDRV7 ;DRIVE 7

1264
1265 ;QUEUE OUTPUT POINTERS
1266
1267 046074 046136 QOUTPT: .WORD QDRV0 ;DRIVE 0
1270 046076 046156 .WORD QDRV1 ;DRIVE 1
046100 046176 .WORD QDRV2 ;DRIVE 2
046102 046216 .WORD QDRV3 ;DRIVE 3
046104 046236 .WORD QDRV4 ;DRIVE 4
046106 046256 .WORD QDRV5 ;DRIVE 5
046110 046276 .WORD QDRV6 ;DRIVE 6
046112 046316 .WORD QDRV7 ;DRIVE 7

1271
1272 046114 046136 QSTART: .WORD QDRV0 ;DRIVE 0 START ADDRESS
1273 046116 046156 QSTOP: .WORD QDRV1 ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
1274 046120 046176 .WORD QDRV2 ;STOP DRIVE 1--START DRIVE 2
1275 046122 046216 .WORD QDRV3 ;STOP DRIVE 2--START DRIVE 3
1276 046124 046236 .WORD QDRV4 ;STOP DRIVE 3--START DRIVE 4
1277 046126 046256 .WORD QDRV5 ;STOP DRIVE 4--START DRIVE 5
1278 046130 046276 .WORD QDRV6 ;STOP DRIVE 5--START DRIVE 6
1279 046132 046316 .WORD QDRV7 ;STOP DRIVE 6--START DRIVE 7
1280 046134 046336 .WORD QTERM ;STOP DRIVE 7
1281
1282 ;DRIVE REQUEST QUEUES
1283
1286 046136 QDRV0: .BLKW 10
046156 QDRV1: .BLKW 10
046176 QDRV2: .BLKW 10
046216 QDRV3: .BLKW 10
046236 QDRV4: .BLKW 10
046256 QDRV5: .BLKW 10
046276 QDRV6: .BLKW 10
046316 QDRV7: .BLKW 10
1287 046336 QTERM=.
1288
1289 ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
1290
1291 ;CALL
1292 JSR PC,CLRQUE
1293
1294 046336 104412 CLRQUE: SAVREG ;SAVE R0 - R5
1295 046340 012702 046044 MOV #QCNT,R2 ;ZERO THE QUEUE COUNTS
1296 046344 005022 CLR (R2)+ ;DRIVES 0 & 1
1297 046346 005022 CLR (R2)+ ;DRIVES 2 & 3

```

```

1298 046350 005022          CLR      (R2)+      ;DRIVES 4 & 5
1299 046352 005022          CLR      (R2)+      ;DRIVES 6 & 7
1300 046354 012703 000010    MOV      #8,R3      ;MOVE THE STARTING
1301 046360 012701 046114    MOV      #QSTART,R1 ;ADDRESS OF THE QUEUE INTO
1302 046364 012122          1$: MOV      (R1)+,(R2)+ ;THE QUEUE INPUT POINTER
1303 046366 005303          DEC      R3
1304 046370 001375          BNE      1$
1305 046372 012703 000010    MOV      #8,R3      ;MOVE THE STARTING ADDRESS
1306 046376 012701 046114    MOV      #QSTART,R1 ;OF THE QUEUE INTO THE
1307 046402 012122          2$: MOV      (R1)+,(R2)+ ;QUEUE OUTPUT POINTER
1308 046404 005303          DEC      R3
1309 046406 001375          BNE      2$
1310 046410 104413          RESREG
1311 046412 000207          RTS      PC      ;RESTORE R0 - R5
1312
1313          ;EMPTY THE QUEUE SPECIFIED BY R1
1314
1315          ;CALL
1316          MOV      DRVNUM,R1      ;DRIVE NUMBER TO R1
1317          JSR      PC,EMPTYQ
1318
1319 046414 105061 046044    EMPTYQ: CLR      QCNT(R1)      ;CLEAR NUMBER OF ITEMS IN QUEUE
1320 046420 006301          ASL      R1
1321 046422 016161 046054 046074    MOV      QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER-INPUT POINTER
1322 046430 006201          ASR      R1
1323 046432 000207          RTS      PC
1324
1325          ;ROUTINE TO PUT A REQUEST IN QUEUE
1326
1327          ;CALL
1328          MOV      #DRVNUM,R1      ;DRIVE NUMBER
1329          MOV      #DPB,R2      ;ADDRESS OF PARAMETER BLOCK
1330          JSR      R0,DRVQUE      ;GO PUT REQUEST IN QUEUE
1331          RETURN1      ;RETURN HERE IF QUEUE IS FULL
1332          RETURN2      ;RETURN HERE IF REQUEST IS IN QUEUE
1333
1334 046434 122761 000010 046044    DRVQUE: CMPB      #10,QCNT(R1)      ;IS QUEUE FULL?
1335 046442 001421          BEQ      2$      ;BR IF YES-TAKE RETURN1
1336 046444 105261 046044          INCB      QCNT(R1)      ;INCREMENT QUEUE COUNT
1337 046450 006301          ASL      R1
1338 046452 010271 046054          MOV      R2,QINPT(R1)      ;PUT THIS REQUEST IN QUEUE
1339 046456 062761 000002 046054          ADD      #2,QINPT(R1)      ;UPDATE THE QUEUE POINTER
1340 046464 026161 046054 046116    CMP      QINPT(R1),QSTOP(R1)      ;TIME TO RESET THE POINTER
1341 046472 001003          BNE      1$      ;BR IF NO
1342 046474 016161 046114 046054          MOV      QSTART(R1),QINPT(R1) ;YES--RESET POINTER
1343 046502 006201          1$: ASR      R1
1344 046504 005720          TST      (R0)+      ;TAKE RETURN 2
1345 046506 000200          2$: RTS      R0      ;RETURN TO USER
1346
1347          ;ROUTINE TO GET THE 'DPB' ADDRESS OF NEXT REQUEST IN QUEUE
1348
1349          ;CALL
1350          MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
1351          JSR      PC,GETREQ      ;GO GET THE REQUEST
1352          RETURN      ;R2='DPB' ADDRESS OF THE REQUEST
1353          ;R2=0 IF NO REQUEST IN QUEUE
1354

```

```

1355 046510 005002
1356 046512 105761 046044
1357 046516 001404
1358 046520 006301
1359 046522 017102 046074
1360 046526 006201
1361 046530 000207
1362
1363
1364
1365
1366
1367
1368
1369
1370 046532 105361 046044
1371 046536 006301
1372 046540 017102 046074
1373 046544 005071 046074
1374 046550 062761 000002 046074
1375 046556 026161 046074 046116
1376 046564 001003
1377 046566 016161 046114 046074
1378 046574 006201
1379 046576 000207
1380
1382
1383
1384
1385
1386
1387
1388
1389
1390 046600 012037 046614
1391 046604 013746 001222
1392 046610 005037 001222
1393 046614 000000
1394 046616 012637 001222
1395 046622 000200

GETREQ: CLR R2
        TSTB QCNT(R1) ;IS THERE ANY REQUEST IN QUEUE?
        BEQ 2$ ;NO
1$: ASL R1
        MOV @QOUTPT(R1),R2 ;PICKUP 'DPB' POINTER FOR THIS DRIVE
        ASR R1
2$: RTS PC ;RETURN TO USER

;ROUTINE TO 'POP' THE REQUEST FROM QUEUE
;CALL
;MOV #DRVNUM,R1 ;DRIVE NUMBER TO R1
;JSR PC,POPQUE ;CALL TO REMOVE REQUEST
;RETURN ;R2=ADDRESS OF DPB REMOVED

POPQUE: DECB QCNT(R1) ;DECREMENT QUEUE COUNT
        ASL R1
        MOV @QOUTPT(R1),R2 ;GET THE 'DPB' POINTER
        CLR @QOUTPT(R1) ;REMOVE DPB ADDRESS FROM THE QUEUE
        ADD #2,QOUTPT(R1) ;UPDATE THE QUEUE POINTER
        CMP QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
        BNE 1$ ;NO--BR TO EXIT
        MOV QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
1$: ASR R1
    RTS PC ;RETURN TO USER

;ROUTINE TO SAVE THE CONTENTS OF '$ESCAPE' WHEN THE DRIVER
;REPORTS AN ERROR DIRECTLY.
;CALL
;JSR ERROR,R0,$ES.SAV ;THE ERROR CALL
;RETURN N ;THE RETURN IS PAST THE ERROR CALL

ES.SAV: MOV (R0)+,1$ ;GET THE ERROR CALL
        MOV $ESCAPE,-(SP) ;SAVE THE ADDRESS IN '$ESCAPE'
        CLR $ESCAPE ;CLEAR THE ESCAPE RETURN
1$: .WORD 0 ;THE ERROR CALL IS MOVED HERE
        MOV (SP)+,$ESCAPE ;RESTORE THE ESCAPE ADDRESS
        RTS R0 ;RETURN
  
```

```

1
2
3
4
5
6
7
8
9
10
11
12
13 046624 005737 001324
14 046630 001427
15 046632 005037 001324
16 046636 012700 001502
17 046642 012703 046776
18 046646 011004
19 046650 004037 035542
20 046654 000402
21 046656 000767
22 046660 000412
23 046662 010420
24 046664 012703 047006
25 046670 011004
26 046672 004037 035542
27 046676 000402
28 046700 000756
29 046702 000401
30 046704 010420
31 046706 010410
32 046710 013701 000004
33 046714 012737 046750 000004
34 046722 005777 132554
35 046726 010137 000004
36 046732 012700 001502
37 046736 012701 040650
38 046742 012021
39 046744 012021
40 046746 000207
41 046750 010137 000004
42 046754 022626
43 046756 104010
44 046760 005737 000042
45 046764 001724
46 046766 005037 001330
47 046772 000137 021334
48
49 046776 200 122 115 MRMCS1: .ASCIIZ <CRLF>/RMCS1=/
50 047006 200 122 115 MRMVEC: .ASCIIZ <CRLF>/RMVEC-/

.SBTTL GETADR - GET BUS ADDRESS AND VECTOR ADDRESS

;THIS ROUTINE IS USED TO ENSURE THE BUS ADDRESS
;OF THE RH/RM IS SETUP TO READ THE PROPER VALUE.
;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
;REQUIRED.
;NOTE: THIS ROUTINE DESTROYS R0-R4
;CALL
;
;JSR PC,GETADR
;RETURN

GETADR: TST BUSADR ;INPUT FROM TTY REQUESTED?
        BEQ 5$ ;NO--BRANCH
        CLR BUSADR ;YES--CLEAR THE REQUEST FLAG
1$: MOV #RH.ADR,R0 ;FIRST ADDRESS
    MOV #MRMCS1,R3 ;'RMCS1='
    MOV (R0),R4 ;PRESENT RMCS1 ADDRESS
    JSR R0,GETNUM ;GET NEW RMCS1
    BR 2$ ;COMMA
    BR 1$ ;PERIOD
    BR 4$ ;DOUBLE PERIOD
2$: MOV R4,(R0)+ ;SAVE NEW RMCS1
    MOV #MRMVEC,R3 ;'RMVEC='
    MOV (R0),R4 ;PRESENT RH/RM VECTOR ADDRESS
    JSR R0,GETNUM ;GET NEW RMVEC
    BR 3$ ;COMMA
    BR 1$ ;PERIOD
    BR 4$ ;DOUBLE PERIOD
3$: MOV R4,(R0)+ ;SAVE NEW RMVEC
4$: MOV R4,(R0) ;SAVE INPUT
5$: MOV ERRVEC,R1 ;SAVE THE ERROR VECTOR
    MOV #6$,ERRVEC ;SETUP FOR TRAP
    TST @RH.ADR ;CHECK FOR RH/RM
    MOV R1,ERRVEC ;RESTORE ERROR VECTOR
    MOV #RH.ADR,R0 ;FIRST ADDRESS OF NEW PARAMETERS
    MOV #RMADR,R1 ;FIRST ADDRESS OF WHERE TO PUT THEM
    MOV (R0)+,(R1)+ ;BUS ADDRESS
    MOV (R0)+,(R1)+ ;VECTOR ADDRESS
    RTS PC ;RETURN
6$: MOV R1,ERRVEC ;RESTORE ERROR VECTOR
    CMP (SP)+,(SP)+ ;CLEAN OFF THE STACK
    EMT 10
    TST @42 ;IS THERE A MONITOR?
    BEQ 1$ ;NO--GO ASK FOR ADDRESS
    CLR DRVSEL ;FUDGE NO DRIVES SELECTED
    JMP $EOP ;RETURN TO $EOP
  
```

			.SBTTL	DPB (DATA PARAMETER BLOCKS)	
1					
2					
3	047016	000	DPB.A:	.BYTE 0	;(0) DRIVE NUMBER
4	047017	000		.BYTE 0	;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
5	047020	000		.BYTE 0	;(2) COMMAND
6	047021	000		.BYTE 0	;(3) PSEL AND A17 AND A16
7	047022	000000		.WORD 0	;(4) WORD COUNT (MUST BE NEG.)
8	047024	054522		.WORD BUFFER	;(6) BUFFER ADDRESS OR
9					REGISTER TABLE POINTER
10	047026	000		.BYTE 0	;(10) SECTOR ADDRESS OR
11					FIRST REG. INDEX
12	047027	000		.BYTE 0	;(11) TRACK ADDRESS OR
13					LAST REG. INDEX
14	047030	000000		.WORD 0	;(12) CYLINDER ADDRESS
15	047032	047140		.WORD RM.REG	;(14) ERROR TABLE POINTER
16					POINTS TO THE FIRST OF TWENTY
17					LOCATIONS OF WHERE THE DRIVER
18					IS TO STORE THE RH/RM
19					REGISTERS ON AN ERROR. IF LEFT
20					ZERO REGISTERS ARE NOT SAVED.
21	047034	000000		.WORD 0	;(16) STATUS/ERROR INDICATOR
22					;BIT15=1=>ERROR OCCURRED
23					;BIT07=1=>DONE
24					;BIT14-BIT09 AND BIT06-BIT03
25					;INDICATE TYPE OF ERROR
26					
27	047036	000	DPB.B:	.BYTE 0	;(0) DRIVE NUMBER
28	047037	000		.BYTE 0	;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
29	047040	000		.BYTE 0	;(2) COMMAND
30	047041	000		.BYTE 0	;(3) PSEL AND A17 AND A16
31	047042	177776		.WORD -2	;(4) WORD COUNT (MUST BE NEG.)
32	047044	054522		.WORD BUFFER	;(6) BUFFER ADDRESS OR
33					REGISTER TABLE POINTER
34	047046	000		.BYTE 0	;(10) SECTOR ADDRESS OR
35					FIRST REG. INDEX
36	047047	000		.BYTE 0	;(11) TRACK ADDRESS OR
37					LAST REG. INDEX
38	047050	000000		.WORD 0	;(12) CYLINDER ADDRESS
39	047052	047140		.WORD RM.REG	;(14) ERROR TABLE POINTER
40					POINTS TO THE FIRST OF TWENTY
41					LOCATIONS OF WHERE THE DRIVER
42					IS TO STORE THE RH/RM
43					REGISTERS ON AN ERROR. IF LEFT
44					ZERO REGISTERS ARE NOT SAVED.
45	047054	000000		.WORD 0	;(16) STATUS/ERROR INDICATOR
46					;BIT15=1=>ERROR OCCURRED
47					;BIT07=1=>DONE
48					;BIT14-BIT09 AND BIT06-BIT03
49					;INDICATE TYPE OF ERROR
50					
51	047056	000	DPB.C:	.BYTE 0	;(0) DRIVE NUMBER
52	047057	000		.BYTE 0	;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
53	047060	000		.BYTE 0	;(2) COMMAND
54	047061	000		.BYTE 0	;(3) PSEL AND A17 AND A16
55	047062	177776		.WORD -2	;(4) WORD COUNT (MUST BE NEG.)
56	047064	054522		.WORD BUFFER	;(6) BUFFER ADDRESS OR
57					REGISTER TABLE POINTER

58	047066	000	.BYTE	0	;(10) SECTOR ADDRESS OR
59					;FIRST REG. INDEX
60	047067	000	.BYTE	0	;(11) TRACK ADDRESS OR
61					;LAST REG. INDEX
62	047070	000000	.WORD	0	;(12) CYLINDER ADDRESS
63	047072	047140	.WORD	RM.REG	;(14) ERROR TABLE POINTER
64					;POINTS TO THE FIRST OF TWENTY
65					;LOCATIONS OF WHERE THE DRIVER
66					;IS TO STORE THE RH/RM
67					;REGISTERS ON AN ERROR. IF LEFT
68					;ZERO REGISTERS ARE NOT SAVED.
69	047074	000000	.WORD	0	;(16) STATUS/ERROR INDICATOR
70					;BIT15=1=>ERROR OCCURRED
71					;BIT07=1=>DONE
72					;BIT14-BIT09 AND BIT06-BIT03
73					;INDICATE TYPE OF ERROR
74					
75	047076	000	DTADPB: .BYTE	0	;(0) DRIVE NUMBER
76	047077	000	.BYTE	0	;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
77	047100	000	.BYTE	0	;(2) COMMAND
78	047101	000	.BYTE	0	;(3) PSEL AND A17 AND A16
79	047102	000000	.WORD	0	;(4) WORD COUNT (MUST BE NEG.)
80	047104	054522	.WORD	BUFFER	;(6) BUFFER ADDRESS OR
81					;REGISTER TABLE POINTER
82	047106	000	.BYTE	0	;(10) SECTOR ADDRESS OR
83					;FIRST REG. INDEX
84	047107	000	.BYTE	0	;(11) TRACK ADDRESS OR
85					;LAST REG. INDEX
86	047110	000000	.WORD	0	;(12) CYLINDER ADDRESS
87	047112	047140	.WORD	RM.REG	;(14) ERROR TABLE POINTER
88					;POINTS TO THE FIRST OF TWENTY
89					;LOCATIONS OF WHERE THE DRIVER
90					;IS TO STORE THE RH/RM
91					;REGISTERS ON AN ERROR. IF LEFT
92					;ZERO REGISTERS ARE NOT SAVED.
93	047114	000000	.WORD	0	;(16) STATUS/ERROR INDICATOR
94					;BIT15=1=>ERROR OCCURRED
95					;BIT07=1=>DONE
96					;BIT14-BIT09 AND BIT06-BIT03
97					;INDICATE TYPE OF ERROR
98					
99	047116	000	DPB.R: .BYTE	0	;(0) DRIVE NUMBER
100	047117	000	.BYTE	0	;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
101	047120	107	.BYTE	RECAL	;(2) COMMAND
102	047121	000	.BYTE	0	;(3) PSEL AND A17 AND A16
103	047122	000000	.WORD	0	;(4) WORD COUNT (MUST BE NEG.)
104	047124	054522	.WORD	BUFFER	;(6) BUFFER ADDRESS OR
105					;REGISTER TABLE POINTER
106	047126	000	.BYTE	0	;(10) SECTOR ADDRESS OR
107					;FIRST REG. INDEX
108	047127	000	.BYTE	0	;(11) TRACK ADDRESS OR
109					;LAST REG. INDEX
110	047130	000000	.WORD	0	;(12) CYLINDER ADDRESS
111	047132	047140	.WORD	RM.REG	;(14) ERROR TABLE POINTER
112					;POINTS TO THE FIRST OF TWENTY
113					;LOCATIONS OF WHERE THE DRIVER
114					;IS TO STORE THE RH/RM

```

115                                     ;REGISTERS ON AN ERROR. IF LEFT
116                                     ;ZERO REGISTERS ARE NOT SAVED.
117 047134 000000                      .WORD 0                               ;(16) STATUS/ERROR INDICATOR
118                                     ;BIT15=1=>ERROR OCCURRED
119                                     ;BIT07=1=>DONE
120                                     ;BIT14-BIT09 AND BIT06-BIT03
121                                     ;INDICATE TYPE OF ERROR
122 047136 000000                      .WORD 0                               ;SKIP SECTOR ENABLE INDICATOR (ENABLED =1)
123
124                                     ;SAVE RH/RM REGISTERS HERE ON ERROR
125
126 047140 000000                      RM.REG: .WORD 0                       ;RMCS1 (776700) CONTROL & STATUS #1
127 047142 000000                      .WORD 0                               ;RMWC (776702) WORD COUNT
128 047144 000000                      .WORD 0                               ;RMSA (776704) BUS ADDRESS
129 047146 000000                      .WORD 0                               ;RMDA (776706) DESIRED SECTOR/TRACK
130 047150 000000                      .WORD 0                               ;RMCS2 (776710) CONTROL & STATUS #2
131 047152 000000                      .WORD 0                               ;RMDS (776712) DISK STATUS
132 047154 000000                      .WORD 0                               ;RMER1 (776714) ERROR REG. #1
133 047156 000000                      .WORD 0                               ;RMAS (776716) ATTENTION SUMMARY
134 047160 000000                      .WORD 0                               ;RMLA (776720) LOOK AHEAD
135 047162 000000                      .WORD 0                               ;RMDB (776722) DATA BUFFER
136 047164 000000                      .WORD 0                               ;RMMR1 (776724) MAINTAINABILITY
137 047166 000000                      .WORD 0                               ;RMDT (776726) DRIVE TYPE
138 047170 000000                      .WORD 0                               ;RMSN (776730) SERIAL NUMBER
139 047172 000000                      .WORD 0                               ;RMOF (776732) OFFSET
140 047174 000000                      .WORD 0                               ;RMDC (776734) DESIRED CYLINDER
141 047176 000000                      .WORD 0                               ;RMHR (776736) CURRENT CYLINDER
142 047200 000000                      .WORD 0                               ;RMMR2 (776740) ERROR REG #2
143 047202 000000                      .WORD 0                               ;RMER2 (776742) ERROR REG #3
144 047204 000000                      .WORD 0                               ;RMEC1 (776744) ECC POSITION
145 047206 000000                      .WORD 0                               ;RMEC2 (776746) ECC PATTERN
146                                     .EVEN

```

```

1
2
3 047210 122 000 MSG.R: .ASCII2 /R/
4 047212 106 103 000 MSG.FC: .ASCII2 /FC/
5 047215 114 103 000 MSG.LC: .ASCII2 /LC/
6 047220 111 103 000 MSG.IC: .ASCII2 /IC/
7 047223 106 124 000 MSG.FT: .ASCII2 /FT/
8 047226 114 124 000 MSG.LT: .ASCII2 /LT/
9 047231 111 124 000 MSG.IT: .ASCII2 /IT/
10 047234 106 124 047 MES.FT: .ASCII2 /F"/
11 047240 114 124 047 MES.LT: .ASCII2 /LT"/
12 047244 111 124 047 MES.IT: .ASCII2 /IT"/
13 047250 106 123 000 MSG.FS: .ASCII2 /FS/
14 047253 114 123 000 MSG.LS: .ASCII2 /LS/
15 047256 120 101 124 MSG.PAT: .ASCII2 /PAT/
16 047262 075 000 MSG.EQ: .ASCII2 /=/
17 047264 200 103 117 MSG.CS: .ASCII2 <CRLF>/CONTROL SWITCHES=/
18
19 047307 040 057 040 SLASH: .ASCII2 @ / @
20 047313 200 125 116 UNSTAT: .ASCII2 <CRLF>/UNIT STATUS:/
21 047331 040 117 106 UNTOFF: .ASCII2 / OFFLINE/
22 047342 040 117 116 UNTON: .ASCII2 / ONLINE/
23 047352 040 116 117 NOTPRS: .ASCII2 / NOT PRESENT/
24 047367 040 125 116 NOTSAF: .ASCII2 / UNSAFE/
25 047377 040 116 117 NOTRM: .ASCII2 @ NOT AN RM05/3/2@
26 047420 040 111 123 LODEV: .ASCII2 / IS LOAD DEVICE/
27 047440 122 115 060 $RM02: .ASCII2 /RM02/
28 047445 122 115 060 $RM03: .ASCII2 /RM03/
29 047452 122 115 060 $RM05: .ASCII2 /RM05/
30 047457 200 104 122 DRIVES: .ASCII2 <CRLF>/DRIVE(S) TO BE TESTED, /
31 047510 116 117 116 NONE: .ASCII2 /NONE/
32 047515 054 040 000 COMMA: .ASCII2 /, /
33 047520 200 116 117 NOCLOK: .ASCII2 <CRLF>/NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED/
34 047605 115 102 101 SERIAL: .ASCII2 @MBA S/N: @
35 047617 200 124 105 MSGTST: .ASCII2 <CRLF>/TEST/
36 047625 200 104 122 MSDRIV: .ASCII2 <CRLF>/DRIVE/
37 047634 040 104 122 DROP: .ASCII2 / DROPPED/
38 047645 105 130 103 EXCEED: .ASCII2 /EXCEEDED MAXIMUM ERROR LIMIT/<CRLF>
39 047703 200 116 117 NODRVS: .ASCII2 <CRLF>/NO DRIVES TO TEST/<CRLF>
40 047727 200 116 117 NOTEST: .ASCII2 <CRLF>/NO TESTS SPECIFIED/<CRLF>
41
42 047754 200 012 122 ROTATE: .ASCII2 <CRLF><LF>/ROTATIONAL SPEED TIMES/
43 050005 200 012 117 ONECYL: .ASCII2 <CRLF><LF>/ONE CYLINDER SEEK TIMES/<CRLF>/ * FORWARD/
44 050052 200 012 101 AVERAGE: .ASCII2 <CRLF><LF>/AVERAGE SEEK TIMES/<CRLF>/ * FORWARD/
45 050112 200 012 115 MXSEEK: .ASCII2 <CRLF><LF>/MAXIMUM SEEK TIMES/
46 050136 200 040 052 FWD: .ASCII2 <CRLF>/ * FORWARD/
47 050152 200 040 052 REV: .ASCII2 <CRLF>/ * REVERSE/
48
49 050166 200 115 111 MSGMIN: .ASCII2 <CRLF>/MIN=/
50 050174 200 115 101 MSGMAX: .ASCII2 <CRLF>/MAX=/
51 050202 200 101 126 MSGAVG: .ASCII2 <CRLF>/AVG=/
52 050210 060 040 125 MSGOUS: .ASCII2 /O US/
53 050215 040 102 105 MBELOW: .ASCII2 / BELOW THE MINIMUM OF /
54 050244 040 101 102 MABOVE: .ASCII2 / ABOVE THE MAXIMUM OF /
55 050273 040 123 105 MSGSEA: .ASCII2 / SEARCHES TIMED/
56 050313 040 123 105 MSGNUM: .ASCII2 / SEEKS TIMED/
57 050330 040 116 117 MSGNON: .ASCII2 / NOT TIMED/
  
```

58	050343	040			BLNKS4:	.ASCII	/ /
59	050344	040			BLNKS3:	.ASCII	/ /
60	050345	040			BLNKS2:	.ASCII	/ /
61	050346	040	000		BLNKS1:	.ASCII	/ /
62	050350	101	114	114	MSG7XA:	.ASCIZ	@ALLOWABLE ROTATIONAL SPEED LIMITS FOR RM05/3@
63	050425	101	114	114	MSG7XB:	.ASCIZ	/ALLOWABLE ROTATIONAL SPEED LIMITS FOR RM02/
64	050500	101	114	114	MSG10X:	.ASCIZ	/ALLOWABLE ONE CYLINDER SEEK LIMIT/
65	050542	101	114	114	MSG11X:	.ASCIZ	/ALLOWABLE AVERAGE SEEK TIME LIMIT/
66	050604	101	114	114	MSG12X:	.ASCIZ	/ALLOWABLE MAXIMUM (FORWARD) SEEK TIME LIMIT/

				.SBTTL ERROR HEADER (EM) MESSAGES	
1					
2					
3	050660	122	110	040	EM1: .ASCIZ /RH CONTROLLER INTERRUPT OCCURRED (RMAS 0)/
4	050732	125	116	105	EM2: .ASCIZ /UNEXPECTED ATTENTION OCCURRED/
5	050770	115	101	123	EM3: .ASCIZ /MASSBUS PARITY ERROR(MCPE=1)/
6	051025	115	101	123	EM4: .ASCIZ /MASSBUS PARITY ERROR(PAR=1)/
7	051061	101	104	104	EM5: .ASCIZ /ADDRESS PLUG CHANGE BIT SET/
8	051115	122	110	040	EM10: .ASCIZ /RH CONTROLLER FAILED TO RESPOND TO ADDRESSING/
9	051173	104	122	111	EM11: .ASCIZ /DRIVE SELECTED IS NOT ONLINE/
10	051230	111	115	120	EM12: .ASCIZ /IMPROPER HEADER DATA/
11	051255	104	101	124	EM13: .ASCIZ /DATA COMPARE FAILURE/
12	051302	104	111	123	EM17: .ASCIZ /DISK ERROR IN TIMING TEST/
13	051334	103	114	117	EM20: .ASCIZ /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/
14	051403	104	111	123	EM23: .ASCIZ /DISK ERROR DURING SEEK/
15	051432	123	105	105	EM24: .ASCIZ /SEEK NOT COMPLETE WITHIN 120 MS/
16	051472	122	110	057	EM41: .ASCIZ @RH/RM ERROR@
17	051506	106	101	124	EM46: .ASCIZ /FATAL WRITE CHECK ERROR/

				.SBTTL STATUS/ERROR INDICATOR MESSAGES	
1					
2					
3	051536	117	106	106	MSG014: .ASCIZ /OFFLINE OR UNSAFE DRIVE REQUESTED/
4	051600	125	116	114	MSG013: .ASCIZ /UNLOADED DRIVE REQUESTED/
5	051631	120	105	122	MSG012: .ASCIZ /PERSISTENT UNSAFE/
6	051653	120	101	122	MSG011: .ASCIZ /PARITY ERROR OCCURRED/
7	051701	106	101	124	MSG010: .ASCIZ /FATAL PARITY ERROR/
8	051724	123	117	106	MSG009: .ASCIZ /SOFTWARE TIMEOUT ON THIS DRIVE/
9	051763	123	117	106	MSG008: .ASCIZ /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
10	052025	105	122	122	MSG006: .ASCIZ 'ERROR OCCURRED DURING I/O OPERATION'
11	052071	105	122	122	MSG005: .ASCIZ 'ERROR OCCURRED DURING NON-I/O OPERATION'
12	052141	125	116	123	MSG004: .ASCIZ /UNSAFE OCCURRED/
13	052161	101	125	124	MSG003: .ASCIZ /AUTOMATIC RECALIBRATE SEQUENCE OCCURRED/
14	052231	104	122	111	MSG002: .ASCIZ /DRIVE HAS NOT RESPONDED TO PORT REQUEST/
15	052301	104	122	111	MSG001: .ASCIZ /DRIVE HAS BECOME NON-EXISTENT/

```

1
2
3 052337 105 122 122 DH1: .ASCIIZ /ERR PC RMAS/
4 052354 105 122 122 DH2: .ASCIIZ /ERR PC DRIVE RMAS RMDS RMER1 RMMR2 RMER2/
5 052441 124 105 123 DH3: .ASCIIZ /TEST ERR PC ADDRESS DATA/
6 052476 124 105 123 DH4: .ASCIIZ /TEST ERR PC ADDRESS GDDATA BDDATA/
7 052545 122 115 103 DH10: .ASCIIZ /RMCS1 ERR PC/
8 052564 104 122 111 DH11: .ASCIIZ /DRIVE ERR PC/
9 052603 124 105 123 DH12: .ASCIIZ /TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR/
10 052672 107 104 103 DH12A: .ASCIIZ /GDCYL GDTRK GDSCTR BDCYL BDTRK BDSCTR/
11 052751 107 104 104 DH13A: .ASCIIZ /GDDAT BDDAT WRDCNT GDADR BDADR/
12 053017 124 105 123 DH17: .ASCIIZ /TEST ERR PC DRIVE RMCS1 RMDS RMER1 RMMR2 RMER2/
13 053114 124 105 123 DH21: .ASCIIZ /TEST ERR PC TST PC DRIVE CYLNR TRACK/
14 053172 107 104 104 DH21A: .ASCIIZ /GDDAT BDDAT WRDCNT SECTOR/
15 053231 124 105 123 DH23: .ASCIIZ /TEST ERR PC DRIVE CYLNR RMCS1 RMCS2 RMDS/
16 053316 122 115 105 DH23A: .ASCIIZ /RMER1 RMMR2 RMER2 RMDC RMHR/
17 053363 124 105 123 DH41: .ASCIIZ /TEST ERR PC TST PC DRIVE/
18 053421 124 105 123 DH42: .ASCIIZ /TEST ERR PC TST PC DRIVE RMCS1 RMCS2 RMDS/
19 053506 122 115 105 DH43A: .ASCIIZ /RMER1 RMMR2 RMER2/
20 053534 122 115 103 DH44A: .ASCIIZ /RMCS1 RMCS2 RMDS RMHR RMDC RMDA/
21 053610 122 115 105 DH44B: .ASCIIZ /RMER1 RMMR2 RMER2/
22 053636 122 115 105 DH45A: .ASCIIZ /RMER1 RMMR2 RMER2 RMWC RMBA RMDB/
23
    .EVEN
  
```

ZRMVBO RM05/3/2 EXT'D DR TST
DATA TABLE (DT)

MACRO V04.00 4-APR-81 11:57:12 PAGE 41

SEQ 0198

.SBTTL DATA TABLE (DT)

1						
2						
3	053714	001132	001204	DT1:	.WORD	\$ERRPC,\$REG3
4	053720	001132	001200	001204	DT2:	.WORD \$ERRPC,\$REG1,\$REG3,RMERRS,RMERRS+2,RMERRS+6,RMERRS+4
5	053736	001212	001132	045326	DT3:	.WORD \$TMP0,\$ERRPC,RD.ADR,RD.WRD
6	053746	001212	001132	045552	DT4:	.WORD \$TMP0,\$ERRPC,WRT.ADR,WRT.WD,RD.WRD
7	053760	001212	001132	001200	DT5:	.WORD \$TMP0,\$ERRPC,\$REG1,\$REG5,RMERRS,RMERRS+2,RMERRS+6,RMERRS+4
8	054000	001502	001132		DT10:	.WORD RH.ADR,\$ERRPC
9	054004	001202	001132		DT11:	.WORD \$REG2,\$ERRPC
10	054010	001212	001132	001176	DT12:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
11	054026	001366	001372	001370	DT12A:	.WORD CYL.DS,TRK.DS,SEC.DS,CYL.RD,TRK.RD,SEC.RD
12	054042	001212	001132	001176	DT13:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
13	054060	001140	001142	001206	DT13A:	.WORD \$GDDAT,\$BDDAT,\$REG4,\$GDADR,\$BDADR
14	054072	001212	001132	001352	DT17:	.WORD \$TMP0,\$ERRPC,CHKDRV,RM.REG,RM.REG+12,RM.REG+14,RM.REG+40,RM.REG+42
15	054112	001212	001132	001176	DT21:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,CYL.DS,TRK.DS
16	054126	001200	001142	001206	DT21A:	.WORD \$REG1,\$BDDAT,\$REG4,\$REG1
17	054136	001212	001132	001352	DT23:	.WORD \$TMP0,\$ERRPC,CHKDRV,CYL.DS,RM.REG,RM.REG+10,RM.REG+12
18	054154	047154	047200	047202	DT23A:	.WORD RM.REG+14,RM.REG+40,RM.REG+42,RM.REG+34,RM.REG+36
19	054166	001212	001132	001176	DT41:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV
20	054176	001212	001132	001176	DT42:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,RM.REG,RM.REG+10,RM.REG+12
21	054214	001212	001132	001176	DT43:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,RM.REG,RM.REG+10,RM.REG+12
22	054232	047154	047200	047202	DT43A:	.WORD RM.REG+14,RM.REG+40,RM.REG+42
23	054240	001212	001132	001176	DT44:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
24	054256	047140	047150	047152	DT44A:	.WORD RM.REG,RM.REG+10,RM.REG+12,RM.REG+36,RM.REG+34,RM.REG+C6
25	054272	047154	047200	047202	DT44B:	.WORD RM.REG+14,RM.REG+40,RM.REG+42
26	054300	001212	001132	001176	DT45:	.WORD \$TMP0,\$ERRPC,\$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
27	054316	047140	047150	047152	DT45A:	.WORD RM.REG,RM.REG+10,RM.REG+12,RM.REG+36,RM.REG+34,RM.REG+06
28	054332	047154	047200	047202	DT45B:	.WORD RM.REG+14,RM.REG+40,RM.REG+42,RM.REG+2,RM.REG+4,RM.REG+22

1
 2
 3 054346 000001
 4 054350 002
 5 054351 000
 6
 7 054352 000001
 8 054354 007
 9 054355 000
 10
 11 054356 000001
 12 054360 004
 13 054361 000
 14
 15 054362 000001
 16 054364 005
 17 054365 000
 18
 19 054366 000001
 20 054370 002
 21 054371 000
 22
 23 054372 000001
 24 054374 002
 25 054375 000
 26
 27 054376 000002
 28 054400 007
 29 054401 160
 30 054402 052672
 31 054404 006
 32 054405 000
 33
 34 054406 000002
 35 054410 007
 36 054411 160
 37 054412 052751
 38 054414 005
 39 054415 004
 40
 41 054416 000000
 42 054420 005
 43 054421 004
 44
 45 054422 000001
 46 054424 010
 47 054425 000
 48
 49 054426 000002
 50 054430 006
 51 054431 060
 52 054432 053172
 53 054434 004
 54 054435 014
 55
 56 054436 000000
 57 054440 004

.SBTTL DATA FORMAT (DF) TABLE

DF1: .WORD 1
 .BYTE 2
 .BYTE 0
 DF2: .WORD 1
 .BYTE 7
 .BYTE 0
 DF3: .WORD 1
 .BYTE 4
 .BYTE 0
 DF4: .WORD 1
 .BYTE 5
 .BYTE 0
 DF10: .WORD 1
 .BYTE 2
 .BYTE 0
 DF11: .WORD 1
 .BYTE 2
 .BYTE 0
 DF12: .WORD 2
 .BYTE 7
 .BYTE 160
 .WORD DH12A
 .BYTE 6
 .BYTE 0
 DF13: .WORD 2
 .BYTE 7
 .BYTE 160
 .WORD DH13A
 .BYTE 5
 .BYTE 4
 DF14: .WORD 0
 .BYTE 5
 .BYTE 4
 DF17: .WORD 1
 .BYTE ^D8
 .BYTE 0
 DF21: .WORD 2
 .BYTE 6
 .BYTE 60
 .WORD DH21A
 .BYTE 4
 .BYTE 14
 DF22: .WORD 0
 .BYTE 4

;NUMBER OF DATA HEADERS
 ;NUMBER OF WORDS IN DATA TABLE
 ;ALL 3 NUMBERS ARE OCTAL

;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

58	054441	014		.BYTE	14	
59						
60	054442	000002	DF23:	.WORD	2	
61	054444	007		.BYTE	7	
62	054445	010		.BYTE	10	
63	054446	053316		.WORD	DH23A	;WORD 4 IS DECIMAL
64	054450	005		.BYTE	5	
65	054451	000		.BYTE	0	
66						
67						
68	054452	000001	DF41:	.WORD	1	
69	054454	004		.BYTE	4	
70	054455	000		.BYTE	0	
71						
72	054456	000003	DF42:	.WORD	1	
73	054460	007		.BYTE	7	
74	054461	000		.BYTE	0	
75						
76	054462	000002	DF43:	.WORD	2	
77	054464	007		.BYTE	7	
78	054465	000		.BYTE	0	
79	054466	053506		.WORD	DH43A	
80	054470	003		.BYTE	3	
81	054471	000		.BYTE	0	
82						
83	054472	000003	DF44:	.WORD	3	
84	054474	007		.BYTE	7	
85	054475	160		.BYTE	160	
86	054476	053534		.WORD	DH44A	
87	054500	006		.BYTE	6	
88	054501	000		.BYTE	0	
89	054502	053610		.WORD	DH44B	
90	054504	003		.BYTE	3	
91	054505	000		.BYTE	0	
92						
93	054506	000003	DF45:	.WORD	3	
94	054510	007		.BYTE	7	
95	054511	160		.BYTE	160	
96	054512	053534		.WORD	DH44A	
97	054514	006		.BYTE	6	
98	054515	000		.BYTE	0	
99	054516	053636		.WORD	DH45A	
100	054520	006		.BYTE	6	
101	054521	000		.BYTE	0	
102						
103			.SBTTL	START OF READ/WRITE BUFFER		
104	054522		BUFFER:			
105						
106		000200	.END	200		

SYMBOL TABLE

ABASE = 000000
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACTDRV 040576
ACTSTR 040577
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV = 000000
AENVM 000000
AFATAL 000000
AMADR1= 000000
AMADR2 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
AOE = 001000
APASS = 000000
APRIOR= 000000
APTCU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPD= 000100
ASWREG= 000000
ATA = 100000
ATABIT 040636
ATESTN= 000000
ATO = 000001
AT1 = 000002
AT2 = 000004
AT3 = 000010

AT4 = 000020
AT5 = 000040
AT6 = 000100
AT7 = 000200
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
AVERAGE 050052
A16 = 000400
A17 = 001000
BADTMO 004576
BAI = 000010
BASFLG 001466
BITS = 001540
BIT0 = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1 = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2 = 000004
BIT3 = 000010
BIT4 = 000020
BIT5 = 000040
BIT6 = 000100
BIT7 = 000200
BIT8 = 000400
BIT9 = 001000
BLNKS1 050346
BLNKS2 050345
BLNKS3 050344
BLNKS4 050343
BPTVEC= 000014
BSE = 100000
BUFFER 054522
BUSADR 001324
BYPASS 001350
CALL.A 030402
CALL.B 030550
CALL.C 030766
CALL.R 032016
CHKDRV 001352
CI1 042126
CI3 042234

CI4 042342
CI5 042720
CI6 042742
CI7 042756
I7B 043000
CI8 043056
CKSCTR 034034
CKSWR = 104407
CK.CHR 040132
CK.DEC 040104
CK.DIG 040206
CK.NUM 040372
CK.OCT 040056
CLKSTA 001342
CLOSE 037776
CLR = 000040
CLRBUF 033766
CLRQUE 046336
CLSWDS 037676
CNTCLR 027334
CNTRLC 001322
COMMA 047515
CONT = 000100
COUNT 033000
CPSAVE 022256
CR = 000015
CRLF = 000200
CYL.DS 001366
CYL.RD 001360
C.SWR 001314
DATCMP 034474
DCK = 100000
DDISP = 177570
DECSEC 033704
DECSK 010110
DELTA 001450
DFLT 002526
DF1 054346
DF10 054366
DF11 054372
DF12 054376
DF13 054406
DF14 054416
DF17 054422
DF2 054352
DF21 054426
DF22 054436
DF23 054442
DF3 054356
DF4 054362
DF41 054452
DF42 054456
DF43 054462
DF44 054472
DF45 054506
DH1 052337
DH10 052545

DH11 052564
DH12 052603
DH12A 052672
DH13A 052751
DH17 053017
DH2 052354
DH21 053114
DH21A 053172
DH23 053231
DH23A 053316
DH3 052441
DH4 052476
DH41 053363
DH42 053421
DH43A 053506
DH44A 053534
DH44B 053610
DH45A 053636
DISPLA 001156
DISPRE 000174
DLT = 100000
DMD = 000001
DORTI 032734
DPB.A 047016
DPB.B 047036
DPB.C 047056
DPB.R 047116
DPINT 040552
DPR = 000400
DPRQS 040562
DRIVES 047457
DROP 047634
DRQ = 004000
DRVACT 040522
DRVCL 031204
DRVCLR= 000111
DRVCL1 031224
DRVINT 041072
DRVMSK 001354
DRVQUE 046434
DRVSEL 001330
DRVSTA 040532
DRVTyp 040542
DRY = 000200
DSWR = 177570
DTADPB 047076
DTE = 010000
DTG = 000020
DT0 020000
DTUW 040634
DT00 = 000001
DT01 = 000002
DT02 = 000004
DT03 = 000010
DT04 = 000020
DT05 = 000040
DT06 = 000100

DT07 = 000200
DT08 = 000400
DT1 = 053714
DT10 054000
DT11 054004
DT12 054010
DT12A 054026
DT13 054042
DT13A 054060
DT17 054072
DT2 053720
DT21 054112
DT21A 054126
DT23 054136
DT23A 054154
DT3 053736
DT4 053746
DT41 054166
DT42 054176
DT43 054214
DT43A 054232
DT44 054240
DT44A 054256
DT44B 054272
DT45 054300
DT45A 054316
DT45B 054332
DT5 053760
DVA = 004000
EBL = 020000
ECH = 000100
ECI = 004000
ECRC = 001000
EECC = 000020
EMPTYQ 046414
EMTVEC= 000030
EM1 050660
EM10 051115
EM11 051173
EM12 051230
EM13 051255
EM17 051302
EM2 050732
EM20 051334
EM23 051403
EM24 051432
EM3 050770
EM4 051020
EM41 051472
EM46 051506
EM5 051061
ERINDX 032170
ERMAX 001316
ERR = 040000
ERRCN 001472
ERROR = 104000
ERRVEC= 000004

ERR.CT 001464	IAE - 002000	MSGB02 052231	OCC - 100000	PKV 001510
ESRC = 004000	IBSAVE 022260	MSGB03 052161	OFD = 000200	PLFS - 002000
ES.SAV 046600	IC 002344	MSGB04 052141	OFFSET= 000115	POPQUE 046532
EXCEED 047645	IE = 000100	MSGB05 052071	OM = 000001	PRM 002334
EXIT.A 014424	ILF = 000001	MSGB06 052025	ONECYL 050005	PRMLMT 002444
EXITG 007462	ILR = 000002	MSGB08 051763	OPE = 020000	PRMMSG 002476
EXIT1 007670	INCCYL 033654	MSGB09 051724	OPI = 020000	PRMPT 002374
EXIT10 013252	INCEC 027216	MSGB10 051701	OPNFLG 001336	PRMO 003132
EXIT11 013516	INCSK 010046	MSGB11 051653	OPNPAT 037364	PRM1 003150
EXIT12 014440	INCTRK 033624	MSGB12 051631	OPNPRM 037162	PRM10 003336
EXIT13 015142	IOTVEC - 000020	MSGB13 051600	OPNTST 037010	PRM11 003354
EXIT14 015702	IR 000100	MSGB14 051536	OPNWDS 037506	PRM12 003374
EXIT15 016452	ISR 043306	MSGMAX 050174	OPN.CT 037014	PRM13 003410
EXIT16 017074	IT 002352	MSGMIN 050166	OPN.N1 037720	PRM14 003420
EXIT17 017540	ITEM41 002254	MSGNON 050330	OPN.N2 037724	PRM15 003430
EXIT2 010132	LC 002342	MSGNUM 050313	OPN.X1 037736	PRM16 003440
EXIT20 020320	LDCMD 030336	MSGSEA 050273	OPN.X2 037742	PRM17 003452
EXIT21 021136	LF = 000012	MSGTST 047617	OPN.1 037020	PRM2 003176
EXIT22 021332	LKS 001526	MSG.CS 047264	OPN.2 037042	PRM20 003462
EXIT3 010352	LKV 001522	MSG.EQ 047262	OPT 041700	PRM21 003516
EXIT4 010764	LODEV 047420	MSG.FC 047212	OR = 000200	PRM22 003526
EXIT5 011210	LODFLT 027704	MSG.FS 047250	PACK = 000123	PRM3 003216
EXIT6 011500	LODPRM 030142	MSG.FT 047223	PAR = 000010	PRM4 003236
EXIT7 012076	LOP.CK 032130	MSG.IC 047220	PAT 002360	PRM5 003256
FC 002340	LPB 001536	MSG.IT 047231	PAT.PT 003536	PRM6 003276
FER = 000020	LPS 001534	MSG.LC 047215	PAT0 003576	PRM7 003316
FILBUF 033730	LPTAVL 001326	MSG.LS 047253	PAT1 003636	PR0 - 000000
FILRAN 035012	LP.AVL 027364	MSG.LT 047226	PAT10 004276	PR1 - 000040
FMT16 = 010000	LS 002356	MSG.PA 047256	PAT11 004336	PR2 - 000100
FS 002354	LSIT = 000002	MSG.R 047210	PAT12 004376	PR3 = 000140
FT 002346	LST - 002000	MSGOUS 050210	PAT13 004436	PR4 = 000200
FWD 050136	LSTRK 001374	MSG10X 050500	PAT14 004476	PR5 = 000240
F1 = 000002	LT 002350	MSG11X 050542	PAT15 004536	PR6 - 000300
F2 = 000004	MABOVE 050244	MSG12X 050604	PAT2 003676	PR7 - 000340
F3 = 000010	MBELOW 050215	MSG7XA 050350	PAT3 003736	PS 177776
F4 = 000020	MCLK = 004000	MSG7XB 050425	PAT4 003776	PSEL - 002000
F5 = 000040	MCPE = 020000	MUR = 001000	PAT5 004036	PSW 177776
GETADR 046624	MCPEMX 040646	MWP = 000010	PAT6 004076	PTRN15 003514
GETNUM 035542	MDF 000100	MXF = 001000	PAT7 004136	PWRVEC= 000024
GETREG- 000141	MES.FT 047234	MXSEEK 050112	PAT8 004176	QCNT 046044
GETREQ 046510	MES.IT 047244	MXSTAL 001462	PAT9 004236	QDRV0 046136
GETSWR 035450	MES.LT 047240	MXWWDW 040656	PDA = 000400	QDRV1 046156
GO - 000001	MI 000004	NBA = 100000	PFECH 022700	QDRV2 046176
GTSWR = 104406	MOC 000400	NC1 002370	PFECH1 022710	QDRV3 046216
GTTST1 036036	MOH 020000	NC2 002372	PFECH2 022772	QDRV4 046236
GTTST2 036122	MOL = 010000	NED = 010000	PFECH3 023024	QDRV5 046256
GTTST3 036464	MPE = 000400	NEM = 004000	PFECH4 023034	QDRV6 046276
GTTST4 036466	MRD - 002000	NOCLOK 047520	PFTSTN 023040	QDRV7 046316
GTTST5 036716	MRMCS1 046776	NODRVS 047703	PGE - 002000	QINPT 046054
GTTST6 036756	MRMVEC 047006	NONE 047510	PGM = 001000	QOUTPT 046074
GT.PRM 035700	MS = 000040	NOOP = 000101	PHA = 000200	QSTART 046114
GT.PR1 035702	MSC = 000002	NOTEST 047727	PIP = 020000	QSTOP 046116
GT.PR2 036032	MSDRIV 047625	NOTPRS 047352	PIRQ = 177772	QTERM = 046336
HCE = 000200	MSEN 010000	NOTRM 047377	PIRQVE= 000240	RANADR 035266
HCI = 002000	MSER - 000200	NOTSAF 047367	PKB 001516	RANCK 035034
HCRC = 000400	MSGAVG 050202	OBCK - 100000	PKC 001520	RANPAT 035232
HT = 000011	MSGB01 052301	OBEN 040000	PKCS 001514	RDCHR = 104410

RDLIN	4411	SAVEFG	040610	START4	004702	TEST12	014116	TST7	011502
RDY	000200	SAVREG-	104412	STATBL	001762	TEST13	014652	*YPDS =	104405
RD.ADR	045326	SC	043520	STKLMT=	177774	TEST14	015354	TYPE =	104401
RD.RM	045302	SCOPE =	000004	STO	044760	TEST15	016114	TYPERR	022262
RD.RM1	045324	SCTRW=	177400	STO1	045014	TEST16	016574	TYPOC =	104402
RD.RM2	045450	SCO	= 000100	STO2	045110	TEST17	017216	*YPON =	104404
RD.RM3	045454	SC1	= 000200	STO3	045140	TEST2	010026	TYPOS =	104403
RD.RM4	045460	SC11	044352	STO5	045164	TEST20	017770	TYPTIM	033240
RD.WRD	045330	SC12	044462	STO6	045172	TEST21	020616	T10	001660
READ =	000171	SC13	044532	STO7	045230	TEST22	021252	T11	001670
READHD=	000173	SC2	= 000400	STO8	045260	TEST3	010270	T12	001700
READIN=	000121	SC3	043570	STO9	045270	TEST4	010524	T7A	001620
RECAL =	000107	SC4	043574	STRTMR	032736	TEST5	011122	T7A1	001640
RELEAS=	000113	SC5	043606	STRT1A	004674	TEST6	011346	T7B	001630
RESREG=	104413	SC6	044002	STRT2A	004716	TEST7	011642	T7B1	001650
RESVEC=	000010	SC6A	044116	ST.CLK	027426	TICKMS	001344	ULDFLG	040600
REV	050152	SC7	044244	ST.LCL	027636	TICKUS	001346	UNLOAD=	000103
REX =	010000	SC8	044322	ST.PCL	027574	TIMER	040614	UNS =	040000
RHVEC	001504	SEARCH=	000131	SVADR	001440	TIM.DN	001414	UNSTAT	047313
RH.ADR	001502	SEC.DS	001370	SVRH70	045654	TIM.PT	001432	UNTOFF	047331
RMADR	040650	SEC.RD	001364	SVSTAT	001356	TIM.UP	001376	JNTON	047342
RMA5	- 000016	SEEK	- 000105	SWR	001154	TKVEC =	000060	UPE	= 020000
RMB4	= 000004	SEEKFG	040612	SWREG	000176	TPB	001532	US1	= 000001
RMCS1	000000	SEKCN	001446	SW0	= 000001	TPS	001530	US2	= 000002
RMCS2	000010	SEY TMR	001444	SW00	= 000001	TPS50	014436	US4	= 000004
RMDA	- 000006	SELDRV=	000145	SW01	= 000002	TPS60	014420	VERIFY	032410
RMDB	- 000022	SERIAL	047605	SW02	= 000004	TPVEC =	000064	VV	= 000100
RMDC	- 000034	SETBUF	034404	SW03	= 000010	TP50	014430	WC	= 000040
RMD5	000012	SETFOR=	000143	SW04	= 000020	TP60	014412	WCE	= 040000
RMDT	- 000026	SETVEC	006060	SW05	= 000040	TRAPVE=	000034	WCEFLG	001434
RMEC1	- 000044	SET.IE	045772	SW06	= 000100	TRCKWC	001452	WCF	= 000040
RMEC2	- 000046	SHUT	025334	SW07	= 000200	TRE	= 040000	WD	= 000010
RMERRS	040512	SKI	- 040000	SW08	= 000400	TRK.DS	001372	WLE	= 004000
RMER1	- 000014	SLASH	047307	SW09	= 001000	TRK.RD	001362	WRCKD	= 000151
RMER2	- 000042	SPTYP	033132	SW1	= 000002	TRNSWT	040572	WRCKHD=	000153
RMHR	- 000036	SP10	001740	SW10	= 002000	TRTVEC=	000014	WRITE	= 000161
RMINIT	040660	SP11	001746	SW11	= 004000	TSTNMS	001332	WRL	004000
RMLA	000020	SP12	001754	SW12	= 010000	TST0	007222	WRTHD	- 000163
RMMR1	- 000024	SP7A	001710	SW13	= 020000	TST1	007464	WRT.AD	045552
RMMR2	000040	SP7A1	001724	SW14	= 040000	TST10	012100	WRT.RM	045462
RMOF	- 000032	SP7B	001716	SW15	= 100000	TST10A	012614	WRT.R1	045546
RMR	= 000004	SP7B1	001732	SW2	= 000004	TST10B	013206	WRT.R2	045640
RMSN	- 000030	SRCHWT	040574	SW3	= 000010	TST11	013254	WRT.R3	045644
RMTMR	044670	SRCH00	032552	SW4	= 000020	TST12	013520	WRT.R4	045650
RMVEC	040652	SRTDRV	006334	SW5	= 000040	TST13	014442	WRT.R5	045652
RMWC	000002	SRTINT	006002	SW6	= 000100	TST14	015144	WRT.WD	045550
RM.REG	047140	SRVCLK	027672	SW7	= 000200	TST15	015704	XXDP	001470
RM05	041406	STACK -	001100	SW8	= 000400	TST16	016454	\$APTHD	001100
ROTATE	047754	STALL	032326	SW9	= 001000	TST17	017076	\$ATYC	026774
RPT	002336	STALL0	001436	TAB.XY=	001114	TST2	007672	\$ATY1	026750
RSTART	006662	STALL1	001454	TAP	- 040000	TST20	017542	\$ATY3	026756
RSTR1	006712	STALL2	001456	TBITVE=	000014	TST21	020322	\$ATY4	026766
RTC	= 000117	STALL3	001460	TD	043350	TST22	021140	\$AUTOB	001150
RTURN	021600	START	004724	TEST0	007372	TST3	010134	\$BASE	001310
R6	- %000006	START1	004670	TEST1	007654	TST4	010354	\$BDADR	001136
R7	= %000007	START2	004712	TEST10	012356	TST5	010766	\$BDDAT	001142
SAVCSW	001320	START3	004660	TEST11	013440	TST6	011212	\$BELL	001224

\$CHARC	023372	\$ERRTB	002014	\$MAMS3	001274	\$REG4	001206	\$TMP2	001216
\$CKSWR	024360	\$ERTTL	001126	\$MAMS4	001300	\$REG5	001210	\$TN =	000023
\$CMTAG	001114	\$ESCAP	001222	\$MBADR	001102	\$RESRE	026014	\$TNPWR	026302
\$CM1 =	000006	\$ETABL	001254	\$MFLG	027212	\$RM02	047440	\$TPB	001166
\$CM2 =	000014	\$ETEND	001314	\$MNEW	025322	\$RM03	047445	\$TPFLG	001173
\$CM3 =	000006	\$FATAL	001236	\$MSGAD	001250	\$RM05	047452	\$TPS	001164
\$CM4 =	000003	\$FFLG	027214	\$MSGGLG	001252	\$RTNAD	021572	\$TRAP	026052
\$CNTLC	025272	\$FILLC	001172	\$MSGTY	001234	\$SAVRE	025756	\$TRAP2	026074
\$CNTLG	025304	\$FILLS	001171	\$MSWR	025311	\$SB2D	026136	\$TRP =	000014
\$CNTLU	025277	\$GDADR	001134	\$MTYP1	001265	\$SCOPE	025356	\$TRPAD	026106
\$CPUOP	001262	\$GDDAT	001140	\$MTYP2	001271	\$SETUP=	000167	\$STM	001104
\$CRLF	001231	\$GET42	021550	\$MTYP3	001275	\$STUP =	177777	\$STNM	001116
\$DBLK	024040	\$GTSWR	024450	\$MTYP4	001301	\$SUPRS	026366	\$TTYIN	025246
\$DB2D	026172	\$HD =	000000	\$MXCNT	025754	\$SVLAD	025704	\$TYPDS	023624
\$DECVL	026352	\$HIBTS	001100	\$NULL	001170	\$SVPC =	000220	\$TYPE	023042
\$DEVCT	001244	\$HINUM	026524	\$NWTST=	000001	\$SWR =	167000	\$TYPEC	023254
\$DEVM	001312	\$ICNT	001120	\$OCNT	023620	\$SWREG	001256	\$TYPEX	023374
\$DIV	026530	\$INTAG	001151	\$OMODE	023622	\$SWRMK=	000000	\$TYPOC	023422
\$DOAGN	021570	\$ITEMB	001130	\$OVER	025740	\$TESTN	001240	\$TYPON	023436
\$DTBL	024030	\$LF	001232	\$PASS	001242	\$TIMES	001220	\$TYPOS	023376
\$ENDAD	021560	\$LFLG	027213	\$PASTM	001106	\$TKB	001162	\$UNIT	001246
\$ENDCT	021514	\$LONUM	026526	\$QUES	001230	\$TKCNT	024050	\$UNITM	001110
\$ENULL	021574	\$LPADR	001122	\$RAND	026426	\$TKINT	024060	\$USWR	001260
\$ENV	001254	\$LPERR	001124	\$RDCHR	024722	\$TKQEN=	024057	\$VECT1	001304
\$ENVM	001255	\$MADR1	001266	\$RDLIN	025012	\$TKQIN	024052	\$VECT2	001306
\$EOP	021334	\$MADR2	001272	\$RDSZ =	000024	\$TKQOU	024054	\$XOFF =	000023
\$EOPCT	021506	\$MADR3	001276	\$REGAD	001174	\$TKQSR	024056	\$XON =	000021
\$ERFLG	001117	\$MADR4	001302	\$REG0	001176	\$TKS	001160	\$XTSTR	025374
\$ERMAX	001131	\$MAIL	001234	\$REG1	001200	\$TKSRV	024130	\$GET4=	000000
\$ERROR	021620	\$MAMS1	001264	\$REG2	001202	\$TMP0	001212	\$OFILL	023621
\$ERRPC	001132	\$MAMS2	001270	\$REG3	001204	\$TMP1	001214	.\$X =	001100

. ABS. 054522 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 62464 WORDS (244 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 70 PAGES
CZRMVB.BIC,CZRMVB/C CZRMVB.DOC,CZRMVB,SYSMAC/M

\$SGET4	17-1	17-1#													
\$OFILL	21-1	21-1#	21-1*	21-1*											
\$40CAT	18-1	24-1													
\$APTHD	5-12	5-12#													
\$ASTAT	32-1	32-1													
\$ATY1	32-1#														
\$ATY3	20-1	32-1#													
\$ATY4	18-1	32-1#													
\$ATYC	32-1	32-1#													
\$AUTOB	6-0#	11-38*	23-1	23-1	23-1										
\$BASE	6-0#														
\$BDADR	6-0#	33-987*	33-:52*	41-13											
\$BD DAT	6-0#	33-917*	33-989*	33-:54*	41-13	41-16									
\$BELL	6-0#	11-42	18-1	18-1	18-1	23-1	23-1	23-1							
\$CHARC	20-1	20-1#	20-1*	20-1*	20-1*										
\$CKSWR	23-1#	26-1	26-1												
\$CM1	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0#	6-0#	
	6-0#	6-0#	6-0#	6-0#	6-0#										
\$CM2	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0	6-0#	6-0#	
	6-0#	6-0#	6-0#	6-0#	6-0#										
\$CM3	6-0	6-0	6-0#												
\$CM4	6-0	6-0	6-0	6-0	6-0	6-0	6-0#	6-0#	6-0#	6-0#					
\$CMTAG	6-0#	11-33	11-33	11-33	11-33	11-33	11-33	11-33							
\$CNTLC	23-1	23-1	23-1	23-1	23-1#										
\$CNTLG	23-1	23-1#													
\$CNTLU	23-1	23-1	23-1#												
\$CPUOP	6-0#														
\$CRLF	6-0#	11-65	11-105	11-150	12-45	12-65	13-46	13-68	18-1	18-1	18-1	19-42	19-58	19-69	
	19-79	19-93	19-100	20-1	20-1	20-1	23-1	23-1	23-1	23-1	33-15	33-692	33-693	33-708	
	33->35														
\$DB2D	27-1	28-1#													
\$DBLK	22-1	22-1	22-1#												
\$DECVL	28-1	28-1#													
\$DEVCT	6-0#														
\$DEVM	6-0#														
\$DIV	14-70	14-336	14-528	31-1#	33-765										
\$DUAGN	17-1	17-1	17-1#												
\$DTBL	22-1	22-1#													
\$ENDAD	5-9	11-38	17-1#	18-1											
\$ENDCT	11-33	12-40*	12-57*	12-66	17-1#										
\$ENULL	17-1	17-1#													
\$ENV	6-0#	11-38	18-1	20-1	32-1	32-1									
\$ENVM	6-0#	11-33	20-1	20-1	32-1										
\$EOP	13-37	16-372	17-1#	23-7	33-23	33-475	35-47								
\$EOPCT	11-33*	12-66*	13-27*	17-1	17-1#	33-22*									
\$ERFLG	6-0#	18-1	18-1	18-1*	24-1	24-1	24-1	24-1	24-1	24-1*	33-432	33-902	33-933	33-:07	
	33-:66														
\$ERMAX	6-0#	11-33*	13-69*	14-44*	14-79*	14-109*	14-141*	14-170*	14-239*	14-274*	14-316*	14-392*	14-543*	15-29*	
	15-154*	15-233*	15-318*	16-20*	16-77*	16-156*	16-287*	16-372*	24-1	24-1	24-1	24-1*			
\$ERROR	11-33	18-1#													
\$ERRPC	6-0#	18-1	18-1	18-1	18-1*	18-1*	19-111	41-3	41-4	41-5	41-6	41-7	41-8	41-9	
	41-10	41-12	41-14	41-15	41-17	41-19	41-20	41-21	41-23	41-26					
\$ERRTB	8-0#	19-37	19-38												
\$ERTTL	6-0#	17-1	17-1	17-1*	18-1	18-1	18-1*								
\$ESCAP	6-0#	11-33*	15-60*	15-175*	15-254*	15-339*	18-1	18-1	18-1	18-1	24-1*	33-242*	33-249*	33-272*	
	33-279*	33-312*	33-319*	33-355*	33-365*	33-386*	33-393*	33-417*	33-447*	33-454*	33-471*	33-579*	33-584*	33-608*	

	33-615*	33-922*	33-927*	33-997*	33-:03*	33-:50*	34-772	34-830	34-832	34-;18	34-;77	34-=81	34-=91	34--92*
SETABL	6-0#													
SETEND	5-12	6-0#												
SFATAL	6-0#	32-1*												
SFFLG	32-1	32-1#	32-1*	32-1*	32-1*									
SFILLC	6-0#	20-1	20-1	20-1										
SFILLS	6-0#	20-1	20-1											
\$GDADR	6-0#	33-988*	33-:53*	41-13										
\$GDDAT	6-0#	33-990*	33-:55*	41-13										
\$GET42	12-53	13-9	17-1#											
\$GTSWR	23-1#	26-1	26-1											
\$HD	4-37	4-37	4-37											
\$HIBTS	5-12#													
\$HINUM	11-33*	14-70	14-331	14-528	16-293*	30-1	30-1	30-1#	30-1*	33-:37	33-:43*	33-:46*	33-:90	33-:06
\$ICNT	6-0#	11-91*	24-1	24-1	24-1	24-1*	24-1*							
\$INTAG	6-0#	23-1	23-1	23-1	23-1	23-1*								
\$ITEMB	6-0#	18-1	18-1	18-1	18-1	18-1	18-1*	18-1*	19-20					
\$LF	6-0#	18-1	18-1	20-1	20-1	23-1	23-1	23-1						
\$LFLG	32-1#	32-1*												
\$LONUM	11-33*	16-294*	30-1	30-1	30-1#	30-1*	33-547	33-:38	33-:42*	33-:45*	33-:89	33-:04	33-:05	
\$LPADR	6-0#	11-33*	14-70*	14-92*	14-111*	14-143*	14-172*	14-241*	14-276*	14-325*	14-394*	14-545*	15-36*	15-60*
	15-161*	15-175*	15-240*	15-254*	15-325*	15-339*	16-26*	16-83*	16-161*	16-289*	16-318*	16-377*	24-1	24-1
	24-1	24-1*	24-1*											
\$LPERR	6-0#	11-33*	14-44*	14-79*	14-109*	14-119*	14-126*	14-141*	14-151*	14-154*	14-170*	14-185*	14-206*	14-239*
	14-250*	14-274*	14-286*	14-316*	14-340*	14-342*	14-392*	14-454*	14-455*	14-543*	15-29*	15-154*	15-233*	15-318*
	16-20*	16-34*	16-38*	16-40*	16-47*	16-53*	16-77*	16-92*	16-102*	16-120*	16-126*	16-156*	16-234*	16-242*
	16-251*	16-287*	16-303*	16-318*	16-328*	16-334*	16-345*	16-351*	16-361*	16-372*	18-1	24-1	24-1	24-1
	24-1*	33-470	33-587											
\$MADR1	6-0#													
\$MADR2	6-0#													
\$MADR3	6-0#													
\$MADR4	6-0#													
\$MAIL	5-12	5-12	6-0#	11-33	11-38	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392
	14-543	15-29	15-154	15-233	15-318	16-20	16-77	16-156	16-287	16-372	18-1	20-1	24-1	
\$MAMS1	6-0#													
\$MAMS2	6-0#													
\$MAMS3	6-0#													
\$MAMS4	6-0#													
\$MBADR	5-12#													
\$MFLG	32-1	32-1#	32-1*	32-1*										
\$MNEW	23-1	23-1#												
\$MSGAD	6-0#	32-1	32-1*											
\$MSGLG	6-0#	32-1*												
\$MSGTY	6-0#	32-1	32-1	32-1*	32-1*									
\$MSWR	23-1	23-1#												
\$MTYP1	6-0#													
\$MTYP2	6-0#													
\$MTYP3	6-0#													
\$MTYP4	6-0#													
\$MXCNT	24-1	24-1	24-1	24-1#										
\$NULL	6-0#	20-1	20-1	20-1										
\$NWTST	14-44	14-44	14-44#	14-44#	14-79	14-79	14-79#	14-79#	14-109	14-109	14-109#	14-109#	14-141	14-141
	14-141#	14-141#	14-170	14-170	14-170#	14-170#	14-239	14-239	14-239#	14-239#	14-274	14-274	14-274#	14-274#
	14-316	14-316	14-316#	14-316#	14-392	14-392	14-392#	14-392#	14-543	14-543	14-543#	14-543#	15-29	15-29
	15-29#	15-29#	15-154	15-154	15-154#	15-154#	15-233	15-233	15-233#	15-233#	15-318	15-318	15-318#	15-318#
	16-20	16-20	16-20#	16-20#	16-77	16-77	16-77#	16-77#	16-156	16-156	16-156#	16-156#	16-287	16-287

	16-287#	16-287#	16-372	16-372	16-372#	16-372#								
\$OCNT	21-1#	21-1*	21-1*											
\$OMODE	21-1	21-1#	21-1*	21-1*	21-1*	21-1*								
\$OVER	24-1	24-1	24-1	24-1#										
\$PASS	6-0#	11-33*	17-1	17-1	17-1*	17-1*								
\$PASTM	5-12#													
\$QUES	6-0#	18-1	18-1	20-1	20-1	23-1	23-1	23-1	23-1					
\$R2A	26-1													
\$RAND	14-70	14-330	14-528	30-1#	33-546	33-:88	33-;03							
\$RDCHR	23-1#	26-1	26-1											
\$RDDEC	26-1													
\$RDLIN	23-1#	26-1	26-1											
\$RDOC*	26-1													
\$RDSZ	23-1	23-1#												
\$REG0	6-0#	19-14*	41-10	41-12	41-15	41-19	41-20	41-21	41-23	41-26				
\$REG1	6-0#	19-15*	41-4	41-7	41-16	41-16								
\$REG2	6-0#	19-16*	41-9											
\$REG3	6-0#	19-17*	41-3	41-4										
\$REG4	6-0#	19-18*	41-13	41-16										
\$REG5	6-0#	19-19*	41-7											
\$REGAD	6-0#	11-40												
\$RESRE	25-1#	26-1												
\$RHEXT	34-164	34-193	34-<16											
\$RMO2	11-139	37-27#												
\$RMO3	11-136	37-28#												
\$RMO5	11-142	37-29#												
\$RTNAD	17-1#													
\$SAVRE	25-1#	26-1	26-1											
\$SB2D	27-1#	33-700	33-705	33-742	33-747	33-749	33-752	33-757	33-759	33-769	33-772	33->62	33-?13	
\$SCOPE	11-33	24-1#												
\$SETUP	4-298	4-298	4-298	4-298	4-298	4-298	4-298#	4-298#	4-298#	4-298#	4-298#	4-298#	4-298#	11-33
	11-33	11-33	11-33	11-33	11-33	11-33	11-33	11-33	11-33	11-33	11-33	11-38	11-38	11-38
	17-1	17-1	18-1	18-1	18-1	18-1	23-1	23-1	23-1	23-1	23-1	24-1		
\$STUP	4-298	4-298	4-298	4-298	4-298	4-298	4-298#	4-298#	4-298#	4-298#	4-298#	4-298#	4-298#	4-298#
	4-298#	4-298#	4-298#	4-298#										
\$SUPRS	29-1#	33-701	33-706	33-742	33-747	33-749	33-752	33-757	33-759	33-769	33-772	33->62	33-?13	
\$SVLAD	24-1	24-1#												
\$SVPC	5-9	5-9#												
\$SWR	4-27#	4-37	4-38	4-38	4-38	4-38	4-38	4-38	4-38	4-38	5-217#	6-0	6-0	6-0
	11-33	11-33	11-33	11-33	11-33	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392
	14-543	15-29	1											

\$TKQSR	23-1	23-1	23-1	23-1#										
\$TKS	6-0#	20-1	20-1	23-1	23-1	23-1	23-1	23-1	23-1*	23-1*	23-1*	23-1*	23-1*	23-1*
\$TKSRV	23-1	23-1#												
\$TMP0	6-0#	19-11*	41-5	41-6	41-7	41-10	41-12	41-14	41-15	41-17	41-19	41-20	41-21	41-23
	41-26													
\$TMP1	6-0#													
\$TMP2	6-0#													
\$TN	4-28#	4-37	14-44	14-44	14-44	14-44	14-44	14-44	14-44	14-44	14-44	14-44	14-44#	14-45
	14-50	14-70	14-79	14-79	14-79	14-79	14-79	14-79	14-79	14-79	14-79	14-79	14-79#	14-80
	14-88	14-96	14-109	14-109	14-109	14-109	14-109	14-109	14-109	14-109	14-109	14-109	14-109#	14-110
	14-114	14-132	14-141	14-141	14-141	14-141	14-141	14-141	14-141	14-141	14-141	14-141	14-141#	14-142
	14-146	14-160	14-170	14-170	14-170	14-170	14-170	14-170	14-170	14-170	14-170	14-170	14-170#	14-171
	14-175	14-226	14-239	14-239	14-239	14-239	14-239	14-239	14-239	14-239	14-239	14-239	14-239#	14-240
	14-244	14-258	14-263	14-274	14-274	14-274	14-274	14-274	14-274	14-274	14-274	14-274	14-274#	14-274#
	14-275	14-279	14-288	14-305	14-316	14-316	14-316	14-316	14-316	14-316	14-316	14-316	14-316	14-316
	14-316#	14-317	14-321	14-364	14-368	14-392	14-392	14-392	14-392	14-392	14-392	14-392	14-392	14-392
	14-392	14-392#	14-393	14-414	14-494	14-530	14-543	14-543	14-543	14-543	14-543	14-543	14-543	14-543
	14-543	14-543	14-543#	14-544	14-552	14-566	15-29	15-29	15-29	15-29	15-29	15-29	15-29	15-29
	15-29	15-29	15-29#	15-30	15-33	15-39	15-85	15-128	15-143	15-154	15-154	15-154	15-154	15-154
	15-154	15-154	15-154	15-154	15-154	15-154#	15-155	15-158	15-164	15-217	15-233	15-233	15-233	15-233
	15-233	15-233	15-233	15-233	15-233	15-233	15-233#	15-234	15-237	15-243	15-299	15-318	15-318	15-318
	15-318	15-318	15-318	15-318	15-318	15-318	15-318	15-318#	15-319	15-322	15-328	15-386	16-20	16-20
	16-20	16-20	16-20	16-20	16-20	16-20	16-20	16-20	16-20#	16-21	16-22	16-63	16-77	16-77
	16-77	16-77	16-77	16-77	16-77	16-77	16-77	16-77	16-77#	16-78	16-79	16-134	16-156	16-156
	16-156	16-156	16-156	16-156	16-156	16-156	16-156	16-156	16-156#	16-157	16-192	16-260	16-287	16-287
	16-287	16-287	16-287	16-287	16-287	16-287	16-287	16-287	16-287#	16-288	16-290	16-314	16-363	16-372
	16-372	16-372	16-372	16-372	16-372	16-372	16-372	16-372#	16-373	16-374	16-389	33-<85	33-=12	33->17
\$TNPWR	28-1	28-1	28-1#											
\$TPB	6-0#	11-45*	18-1*	18-1*	20-1	20-1	20-1*							
\$TPFLG	6-0													

\$XON	20-1	20-1	23-1			
\$XTSTR	24-1#					
.\$ASTA	32-1	32-1				
.\$X	5-12	5-12#				
A16	4-79#					
A17	4-80#					
ABASE	6-0	6-0				
ACDW1	6-0					
ACDW2	6-0					
ACPUOP	6-0	6-0				
ACTDRV	34-93#	34-347*	34-398*	34-691*	34-699*	34-986
ACTSTR	34-99#	34-988*	34-:02*			
ADDW0	6-0					
ADDW1	6-0					
ADDW10	6-0					
ADDW11	6-0					
ADDW12	6-0					
ADDW13	6-0					
ADDW14	6-0					
ADDW15	6-0					
ADDW2	6-0					
ADDW3	6-0					
ADDW4	6-0					
ADDW5	6-0					
ADDW6	6-0					
ADDW7	6-0					
ADDW8	6-0					
ADDW9	6-0					
ADEVCT	6-0	6-0				
ADEVN	6-0	6-0				
AENV	6-0	6-0				
AENVN	6-0	6-0				
AFATAL	6-0	6-0				
AMADR1	6-0	6-0				
AMADR2	6-0	6-0				
AMADR3	6-0	6-0				
AMADR4	6-0	6-0				
AMAMS1	6-0	6-0				
AMAMS2	6-0	6-0				
AMAMS3	6-0	6-0				
AMAMS4	6-0	6-0				
AMSGAD	6-0	6-0				
AMSGLG	6-0	6-0				
AMSGTY	6-0	6-0				
AMTYP1	6-0	6-0				
AMTYP2	6-0	6-0				
AMTYP3	6-0	6-0				
AMTYP4	6-0	6-0				
AOE	4-151#					
APASS	6-0	6-0				
APRIOR	6-0					
APTCSU	20-1	32-1#				
APTENV	18-1	20-1	32-1	32-1#		
APTSIZ	11-33	32-1#				
APTSP0	20-1	32-1	32-1#			
ASWREG	6-0	6-0				
ATO	4-199#					

AT1	4-200#														
AT2	4-201#														
AT3	4-202#														
AT4	4-203#														
AT5	4-204#														
AT6	4-205#														
AT7	4-206#														
ATA	4-138#														
ATABIT	12-29 34-.51	33-10	33-21	33-<29	33-<35	34-145#	34-318	34-410	34-515	34-895	34-923	34-931	34-932	34-954	
ATESTN	6-0	6-0													
AUNIT	6-0	6-0													
AUSWR	6-0	6-0													
AVECT1	6-0	6-0													
AVECT2	6-0	6-0													
AVERAGE	7-0	37-44#													
BADTMO	11-3#	11-35													
BAI	4-97#														
BASFLG	7-0#	14-81*	16-336	16-353	33-372*	33-400*	33-424*								
BIT0	4-71#														
BIT00	4-71	4-71#	7-0	7-0	12-7	13-39	18-1	18-1	24-1	24-1	33-144				
BIT01	4-71	4-71#	7-0	7-0	33-472	33-500	34-390	34-655							
BIT02	4-71	4-71#	7-0	7-0	33-472	33-503									
BIT03	4-71	4-71#	7-0	7-0	33-509	34-849	34-;71								
BIT04	4-71	4-71#	7-0	7-0	33-506	34-885									
BIT05	4-71	4-71#	7-0	7-0	33-509	34-865									
BIT06	4-71	4-71#	7-0	7-0	33-512	34-328	34-381	34-454	34-734	34-<42					
BIT07	4-71	4-71#	7-0	7-0	34-328	34-590	34-711	34-849	34-865	34-885	34-927	34-<03			
BIT08	4-71	4-71#	7-0	33-500	34-328										
BIT09	4-71	4-71#	7-0	18-1	18-1	24-1	24-1	33-515	34-:39						
BIT1	4-71#														
BIT10	4-71#	7-0	18-1	33-472	33-503	33-?88	34-657								
BIT11	4-71#	7-0	24-1	33-156	33-503	34-281	34-417	34-618	34-972						
BIT12	4-71#	7-0	33-472	33-506	34-275	34-309	34-328	34-392	34-425	34-608	34-653	34-669	34-875	34-877	
	34-:11	34-:66	34-<43												
BIT13	4-71#	7-0	18-1	33-472	33-500	34-375	34-820	34-:14							
BIT14	4-71#	7-0	24-1	33-381	33-472	33-506	34-387	34-422	34-805	34-965	34-:75	34-:21	34-:25	34-<40	
BIT15	4-71#	7-0	33-638	33-639	34-375	34-387	34-390	34-392	34-422	34-425	34-618	34-655	34-657	34-734	
	34-849	34-865	34-875	34-885	34-965	34-:39	34-:75	34-:83							
BIT2	4-71#	34-:83													
BIT3	4-71#														
BIT4	4-71#														
BIT5	4-71#														
BIT6	4-71#														
BIT7	4-71#														
BIT8	4-71#														
BIT9	4-71#														
BITS	7-0#	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233	
	15-318	16-20	16-77	16-156	16-220	16-287	16-372	33-<96	33-=30	33-=71	33->31	33-?66			
BLNKS1	33->90	37-61#													
BLNKS2	11-133	19-63	19-89	19-96	19-103	33-746	33-756	33-771	33->57	33->86	37-60#				
BLNKS3	37-59#														
BLNKS4	11-107	37-58#													
BPTVEC	4-71#														
BSE	4-262#	33-254	33-286	33-326	33-370	33-398	33-422	33-521							
BUFFER	14-407	16-33	16-45	16-56	16-90	16-100	16-115	16-191	16-297	16-324	16-333	16-343	16-350	16-360	
	33-568*	33-569	33-571	33-573	33-574	33-575	33-637	33-674	33-841	33-860	33-861	33-885	33-946	33-965	

	33-980	33-:22	33-:39	33-:40	33-:57	36-8	36-32	36-56	36-80	36-104	42-104#			
BUSADR	7-0#	11-15*	11-18*	11-22*	11-25*	35-13	35-15*							
BYPASS	7-0#	13-37*	14-70*	14-92*	14-117*	14-149*	14-182*	14-247*	14-282*	14-325*	14-454*	14-555*	15-60*	15-175*
	15-254*	15-339*	16-26*	16-83*	16-220*	16-292*	33-434	33-904	33-:09	33-:74				
C.SWR	7-0#	12-6	12-9	12-35	13-39	14-182	15-64	15-124	16-299	33-93	33-103	33-113	33-144	33-226
	33-300	33-340	33-435	33-544	33-:61	33-:65	33-:66*							
CALL.A	13-43	13-45	14-70	14-341	15-60	15-175	15-254	15-339	16-381	16-386	33-242#	33-245	33-583	
CALL.B	14-70	14-95	14-121	14-128	14-152	14-156	14-187	14-193	14-208	14-214	14-252	14-254	14-292	14-294
	14-296	14-298	14-300	14-302	14-361	14-558	33-272#	33-275						
CALL.C	14-94	14-557	33-312#	33-315										
CALL.R	33-259	33-291	33-331	33-429	33-447#	33-450								
CHKDRV	7-0#	13-13*	13-17*	13-22	17-1	33-35*	33-36	33-187	41-10	41-12	41-14	41-15	41-17	41-19
	41-20	41-21	41-23	41-26										
CI1	34-444	34-472#												
CI3	34-446	34-496#												
CI4	34-438	34-518#												
CI5	34-494	34-516	34-596#	34-606										
CI6	34-534	34-544	34-546	34-548	34-602#									
CI7	34-377	34-453	34-484	34-488	34-492	34-501	34-510	34-514	34-526	34-533	34-539	34-543	34-557	34-563
	34-567	34-577	34-588	34-605	34-608#	34-715	34-910	34-<15						
CI7B	34-623#	34-916												
CI8	34-626	34-640#	34-769	34-:66										
CK.CHR	33-:91	33-<26	33->66	33->93	33-?18	33-a42#	33-a79	33-a86	33-a91	33-A36				
CK.DEC	33-a20#	33-a50												
CK.DIG	33-<28	33->80	33-a73#											
CK.NUM	33-:93	33-?02	33-?20	33-A19#										
CK.OCT	33-<73	33-<77	33-=00	33-=04	33-a02#	33-A21	33-A31							
CKSCTR	16-44	33-882#												
CKSWR	18-1	18-1	24-1	26-1#										
CLKSTA	7-0#	12-67	14-454	14-454	15-31	15-156	15-235	15-320	33-78*	33-80*	33-86*			
CLOSE	33-?55	33-?60	33-?72#											
CLR	4-99#	14-454	14-454	14-528	33-34	33-604	34-233	34-680						
CLRBUF	16-41	33-858#												
CLRQUE	14-413	34-215	34-679	34-<94#										
CLSWDS	33-?35	33-?39	33-?45#											
CNTCLR	11-97	13-30	15-100	15-109	15-119	15-121	15-191	15-209	15-211	15-268	15-280	15-291	15-293	15-353
	15-365	15-378	15-380	33-33#										
CNTRLC	7-0#	11-19*	11-26*	11-101	12-3									
COMMA	12-61	13-49	33-19	37-32#										
CONT	4-167#													
COUNT	15-111	15-193	15-271	15-283	15-356	15-368	33-653#							
CPSAVE	18-1	18-1	18-1	18-1	18-1	18-1#	18-1*	18-1*	19-111	24-1	24-1	24-1	24-1*	24-1*
CR	4-71#	20-1	20-1											
CRLF	4-71#	11-6	11-38	11-38	11-59	11-70	11-76	11-77	17-1	17-1	17-1	20-1	20-1	33-:59
	33-<23	35-49	35-50	37-17	37-20	37-30	37-33	37-35	37-36	37-38	37-39	37-39	37-40	37-40
	37-42	37-43	37-43	37-44	37-44	37-45	37-46	37-47	37-49	37-50	37-51			
CYL.DS	7-0#	14-454*	33-250*	33-280*	33-320*	33-366*	33-394*	33-418*	33-455*	33-578*	33-616*	33-920*	33-993*	41-10
	41-11	41-12	41-15	41-17	41-23	41-26								
CYL.RD	7-0#	33-573*	41-11											
DATCMP	16-256	33-964#												
DCK	4-157#													
DDISP	4-71#	6-0	11-33											
DECSEC	33-826#													
DECSK	14-127#	14-131												
DELTA	7-0#	14-401*	14-402*	14-403	14-405*	14-528	14-528	14-528*	14-528*	14-528*				
DF1	8-14	42-3#												
DF10	8-83	42-19#												

DF11	8-94	42-23#												
DF12	8-108	42-27#												
DF13	8-122	8-145	42-34#											
DF14	8-131	8-154	42-41#											
DF17	8-165	8-176	42-45#											
DF2	8-25	8-58	42-7#											
DF21	8-190	42-49#												
DF22	8-199	42-56#												
DF23	8-212	8-225	42-60#											
DF3	8-36	42-11#												
DF4	8-47	42-15#												
DF41	8-258	42-68#												
DF42	8-269	42-72#												
DF43	8-282	42-76#												
DF44	8-298	42-83#												
DF45	8-314	8-330	42-93#											
DFLT	9-61#	33-133												
DH1	8-12	40-3#												
DH10	8-81	40-7#												
DH11	8-92	40-8#												
DH12	8-106	8-120	8-143	8-296	8-312	8-328	40-9#							
DH12A	40-10#	42-30												
DH13A	40-11#	42-37												
DH17	8-163	8-174	40-12#											
DH2	8-23	8-56	40-4#											
DH21	8-188	40-13#												
DH21A	40-14#	42-52												
DH23	8-210	8-223	40-15#											
DH23A	40-16#	42-63												
DH3	8-34	40-5#												
DH4	8-45	40-6#												
DH41	8-256	40-17#												
DH42	8-267	8-280	40-18#											
DH43A	40-19#	42-79												
DH44A	40-20#	42-86	42-96											
DH44B	40-21#	42-89												
DH45A	40-22#	42-99												
DISPLA	6-0#	11-33*	11-33*	18-1*	24-1*									
DISPRE	5-1#	11-33												
DLT	4-109#													
DMD	4-161#	4-180#												
DORTI	14-454	15-92	15-182	15-259	15-344	33-625#								
DPB.A	13-32*	13-38*	13-41*	13-42*	13-44*	14-46*	14-319*	14-339*	14-343	14-395*	14-398*	14-411	14-454	14-454
	14-454	14-528	14-528	14-528	14-528*	14-528*	15-60*	15-60*	15-60*	15-175*	15-175*	15-175*	15-254*	15-254*
	15-254*	15-339*	15-339*	15-339*	16-379*	16-380*	16-384*	16-385*	33-244	33-246	33-250	33-250	33-250	33-250
	33-260	33-457	33-582*	36-3#										
DPB.B	13-33*	14-47*	14-48*	14-49*	14-70*	14-70*	14-82*	14-84*	14-86*	14-112*	14-113*	14-118*	14-122*	14-123
	14-125*	14-129*	14-130	14-144*	14-145*	14-150*	14-155*	14-173*	14-174*	14-186*	14-192*	14-207*	14-213*	14-242*
	14-243*	14-251*	14-253*	14-277*	14-278*	14-291*	14-293*	14-295*	14-297*	14-299*	14-301*	14-318*	14-327*	14-337*
	14-339	14-362*	14-363	14-367*	14-546*	14-548*	14-550*	14-563*	14-564	33-228*	33-231*	33-274	33-276	33-280
	33-280	33-280	33-280	33-292	33-295	33-298	36-27#							
DPB.C	13-34*	14-83*	14-85*	14-87*	14-547*	14-549*	14-551*	14-559*	14-560	14-562*	33-229*	33-232*	33-314	33-316
	33-320	33-320	33-320	33-320	33-332	33-335	33-338	36-51#						
DPB.R	33-449	33-451	33-455	33-455	33-455	33-455	36-99#							
DPINT	34-61#	34-244	34-247	34-357	34-789	34-816	34-955	34-957*	34-:30	34-:53	34-:59	34-:69*		
DPR	4-131#													
DPROS	34-70#	34-363	34-411*	34-448*	34-628*	34-645	34-663*	34-792	34-:32	34-:55	34-:61	34-:79*		

DRIVES	12-39	37-30#												
DROP	33-18	37-37#												
DRQ	4-222#													
DRVACT	14-482*	34-30#	34-371	34-551*	34-599*	34-630*	34-643	34-662*	34-704*	34-812	34-847	34-879*	34-887*	34-907
	34-922*	34-:44*												
DRVCAL	16-36	16-39	16-43	16-52	16-58	16-95	16-106	16-122	16-128	16-239	16-244	16-253	16-304	16-329
	16-335	16-346	16-352	16-362	33-355#	33-359								
DRVCL1	14-488	33-361#	33-362											
DRVCLR	4-277#	14-475												
DRVINT	34-235	34-269#	34-359	34-944	34-958									
DRVMSK	7-0#	13-14*	13-15	13-18*	13-26									
DRVQUE	34-366	34-379	34-:34#											
DRVSEL	7-0#	12-19*	12-29*	12-42	13-5	13-15	13-26*	23-6*	33-10	33-21*	33-<22*	33-<29*	33-<35*	35-46*
DRVSTA	11-108	12-23	13-23	34-40#	34-225*	34-226*	34-227*	34-228*	34-238*	34-270*	34-280*	34-324*	34-330*	34-353
	34-361	34-385	34-419	34-423	34-552*	34-683*	34-795	34-803	34-818	34-873*	34-880*	34-896	34-960	34-:70*
DRVSTP	11-111	11-135	15-69	15-78	33-199	34-52#	34-271*	34-287*	34-292*	34-297*	34-302*	34-388	34-684*	
DRY	4-130#													
DSWR	4-71#	6-0	11-33											
DT00	4-213#													
DT01	4-214#													
DT02	4-215#													
DT03	4-216#													
DT04	4-217#													
DT05	4-218#													
DT06	4-219#													
DT07	4-220#													
DT08	4-221#													
DT1	8-13	41-3#												
DT10	8-82	41-8#												
DT11	8-93	41-9#												
DT12	8-107	41-10#												
DT12A	41-11#													
DT13	8-121	8-144	41-12#											
DT13A	8-130	8-153	41-13#											
DT17	8-164	8-175	41-14#											
DT2	8-24	8-57	41-4#											
DT21	8-189	41-15#												
DT21A	8-198	41-16#												
DT23	8-211	8-224	41-17#											
DT23A	41-18#													
DT3	8-35	41-5#												
DT4	8-46	41-6#												
DT41	8-257	41-19#												
DT42	8-268	41-20#												
DT43	8-281	41-21#												
DT43A	41-22#													
DT44	8-297	41-23#												
DT44A	41-24#													
DT44B	41-25#													
DT45	8-313	8-329	41-26#											
DT45A	41-27#													
DT45B	41-28#													
DT5	41-7#													
DTADPB	13-35*	14-352*	14-355	14-358*	14-360*	14-396*	14-397*	14-399*	14-406*	14-407*	14-465*	14-468	14-471*	14-473*
	14-474	14-479*	14-480	14-486	14-528	14-528*	15-63	15-100	15-109	15-119	15-177*	15-178*	15-184	15-191
	15-196*	15-197	15-201*	15-202	15-209	15-268	15-280	15-291	15-353	15-365	15-378	16-29*	16-30*	16-31*
	16-32*	16-33*	16-35*	16-37*	16-42*	16-48*	16-49*	16-50*	16-51*	16-54*	16-55*	16-56*	16-57*	16-86*

	16-87*	16-88*	16-89*	16-90*	16-94*	16-96*	16-100*	16-104*	16-105*	16-107*	16-113*	16-115*	16-118*	16-121*
	16-124*	16-125*	16-127*	16-129	16-190*	16-191*	16-220*	16-220*	16-221*	16-226*	16-235*	16-243*	16-252*	16-295*
	16-296*	16-297*	16-298*	16-302*	16-305*	16-306	16-308*	16-309	16-313*	16-321	16-322	16-323*	16-325*	16-332*
	16-333*	16-340*	16-341*	16-342*	16-343*	16-349*	16-350*	16-357*	16-358*	16-359*	16-360*	33-358	33-361	33-366
	33-366	33-366	33-366	33-377	33-379	33-385*	33-388	33-390	33-394	33-394	33-394	33-394	33-403*	33-405
	33-407	33-411	33-413	33-418	33-418	33-418	33-418	33-430	33-605*	33-606*	33-607*	33-610	33-612	33-616
	33-616	33-616	33-616	33-920	33-921	33-947	33-966	33-993	33-994	33-995	33-;44*	33-;45*	33-;46*	36-75#
DTE	4-154#													
DTG	4-184#													
DTO	4-193#													
DTUW	14-481*	34-139#	34-221	34-440	34-493*	34-637*	34-648	34-664	34-666*	34-677*	34-694	34-705*	34-:35	34-:47*
	34-:19													
DVA	4-124#													
EBL	4-174#													
ECH	4-148#													
ECI	4-245#													
ECRC	4-170#													
EECC	4-165#													
EM1	8-11	38-3#												
EM10	8-80	38-8#												
EM11	8-91	38-9#												
EM12	8-105	38-10#												
EM13	8-119	8-142	8-187	38-11#										
EM17	8-162	38-12#												
EM2	8-22	38-4#												
EM20	8-173	38-13#												
EM23	8-209	38-14#												
EM24	8-222	38-15#												
EM3	8-33	38-5#												
EM4	8-44	38-6#												
EM41	8-255	8-266	8-279	8-295	8-311	38-16#								
EM46	8-327	38-17#												
EM5	8-55	38-7#												
EMPTYQ	34-627	34-682	34-735	34-:84	34-=:19#									
EMTVEC	4-71#	11-33*	11-33*											
ERINDX	33-250	33-280	33-320	33-366	33-394	33-418	33-455	33-494#	33-616					
ERMAX	7-0#	33-13												
ERR	4-137#	14-454	15-98	15-107	15-189	15-266	15-278	15-351	15-363					
ERR.CT	7-0#	13-69	16-20	16-77	16-156	16-287	33-432	33-902	33-933	33-:07	33-:66			
ERRCN	7-0#	11-86*	11-89*	11-89*	11-89*	11-89*	11-89*	11-89*	11-89*	33-12*	33-13			
ERROR	4-71#													

INCEC	19-41	33-6#																
INCSK	14-120#	14-124																
INCTRK	16-257	33-794#																
IOTVEC	4-71#	11-33*	11-33*															
IR	4-100#																	
ISR	14-454	15-123	15-213	15-295	15-382	33-602	34-230	34-691#										
IT	9-12#	33-796																
ITEM41	8-247#	19-37																
LC	9-8#	14-49	14-70	14-70	14-87	14-123	14-125	14-158	14-203	14-205	14-213	14-249	14-257	14-284				
	14-328	14-333	14-401	14-493	14-528	14-528	14-560	14-564	15-197	15-261	15-346	16-222	16-309	16-379				
	33-811	33-814	33-816	33-;34														
LDCMD	13-36	33-226#																
LF	4-71#	17-1	20-1	20-1	37-42	37-43	37-44	37-45										
LKS	7-0#	33-85	33-117*															
LKV	7-0#	14-454	14-454	33-115*	33-116*													
LGDEV	11-130	37-26#																
LODFLT	12-10	12-18	33-130#															
LODPRM	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233	15-318				
	16-20	16-77	16-156	16-287	16-372	33-180#												
LOP.CK	33-261	33-293	33-333	33-378	33-431	33-458	33-468#											
LP.AVL	11-90	33-46#																
LPB	7-0#	18-1																
LPS	7-0#	18-1	33-49															
LPTAVL	7-0#	18-1	33-46*	33-50*														
LS	9-14#	14-83	16-164															
LSIT	4-162#																	
LST	4-133#																	
LSTRK	7-0#	16-98	16-109	16-129	16-132	16-220	16-224	16-306	33-201*	33-213*	33-;16							
LT	9-11#	14-85	14-363	33-794	33-797	33-799												
MABOVE	33-758	37-54#																
MBELOW	33-748	37-53#																
MCLK	4-191#																	
MCPE	4-82#																	
MCPEMX	34-157#	34-;01	34-;47															
MDF	4-186#																	
MES.FT	9-52	37-10#																
MES.IT	9-54	37-12#																
MES.LT	9-53	37-11#																
MI	4-182#																	
MOC	4-188#																	
MOH	4-223#																	
MOL	4-135#																	
MPE	4-102#																	
MRD	4-190#																	
MRMCS1	35-17	35-49#																
MRMVEC	35-24	35-50#																
MS	4-185#																	
MSC	4-181#																	
MSDRIV	13-47	33-16	37-36#															
MSEN	4-192#																	
MSER	4-187#																	
MSG.CS	33-;60	37-17#																
MSG.EQ	33->61	33->88	33-?14	37-16#														
MSG.FC	9-46	37-4#																
MSG.FS	9-55	37-13#																
MSG.FT	9-49	37-7#																
MSG.IC	9-48	37-6#																

[illegible]

[illegible]

[illegible]

RANCK	16-338	16-355	33-:37#	33-:85#											
RANPAT	16-326	33-:24	33-:44	41-5											
RD.ADR	34-:03*	34-:04*	34-:06#	34-537	34-561	34-575	34-713	34-767	34-841	34-851	34-867	34-:22	34-:01#	34-:54	
RD.RM	34-283	34-313	34-319	34-537	34-561	34-575	34-713	34-767	34-841	34-851	34-867	34-:22	34-:01#	34-:54	
	34-:68	34-:74	34-<07												
RD.RM1	34-:05#	34-:33													
RD.RM2	34-:01*	34-:32#													
RD.RM3	34-:12	34-:24	34-:34#												
RD.RM4	34-:17	34-:36#													
RD.WRD	34-:07#	34-:08	41-5	41-6											
RDCHR	23-1	26-1#													
RDLIN	26-1#	33-:89	33-<24	33-<40	33->64	33->91	33-?16								
RDY	4-78#														
READ	4-288#	16-42	16-252	16-332	16-349										
READHD	4-289#	33-228	33-229	33-295	33-335										
READIN	4-281#														
RECAL	4-276#	13-44	13-44	14-46	14-46	33-582	33-607	36-101							
RELEAS	4-278#														
RESREG	15-100	15-109	15-119	15-191	15-209	15-268	15-280	15-291	15-353	15-365	15-378	19-105	26-1#	28-1	
	33-640	33-849	33-872	33-906	33-954	33-976	33-<36	33->21	33-?36	33-?40	33-?64	33-?90	34-252	34-393	
	34-396	34-458	34-686	34-698	34-:01	34-<26	34-=10								
RESVEC	4-71#														
REV	7-0	7-0	7-0	37-47#											
REX	4-173#														
RH.ADR	7-0#	35-16	35-34	35-36	41-8										
RHVEC	7-0#														
RM.REG	13-52	33-252	33-254	33-257	33-282	33-284	33-286	33-289	33-322	33-324	33-326	33-329	33-368	33-370	
	33-381	33-396	33-398	33-420	33-422	33-427	33-519	33-521	33-826	36-15	36-39	36-63	36-87	36-111	
	36-126#	41-14	41-14	41-14	41-14	41-14	41-17	41-17	41-17	41-18	41-18	41-18	41-18	41-18	
	41-20	41-20	41-20	41-21	41-21	41-21	41-22	41-22	41-22	41-24	41-24	41-24	41-24	41-24	
	41-24	41-25	41-25	41-25	41-27	41-27	41-27	41-27	41-27	41-27	41-28	41-28	41-28	41-28	
	41-28	41-28													
RMOS	33-243	33-273	33-313	33-357	33-387	33-404	33-410	33-448	33-609	34-345#					
RMADR	14-320	14-412	33-7	33-33	33-603	34-162#	34-232	34-352	34-475	34-496	34-518	34-693	34-:20	34-:03	
	34-:09	34-:13	34-:27	34-:60	34-:64	34-:96	34-<37	35-37							
RMAS	34-180#	34-318*	34-765	34-895*	34-931*	34-954*	34-:50								
RMBA	14-477*	34-175#													
RMCS1	14-454*	14-475*	14-486*	15-262*	15-274*	15-347*	15-359*	34-173#	34-274*	34-281	34-:07	34-416*	34-417	34-491	
	34-513	34-556	34-576	34-587	34-604	34-625	34-714	34-768	34-827	34-858	34-903	34-972	34-:55		
RMCS2	14-343*	14-454*	14-454*	14-454*	14-454*	14-456*	14-528*	14-528*	33-8	33-34*	33-36*	33-604*	34-177#	34-233*	
	34-273*	34-275	34-365*	34-415*	34-476*	34-497*	34-519*	34-608	34-653	34-669	34-680*	34-712*	34-839*	34-939*	
	34-953*	34-:21*	34-:10	34-:65	34-:97*	34-<03	34-<38*	34-<43							
RMDA	14-478*	15-60*	33-619*	33-826	34-176#	34-483	34-509	34-525							
RMDB	34-182#	34-<01													
RMDC	14-454*	15-60*	15-184*	15-261*	15-273*	15-346*	15-358*	33-620*	34-187#	34-487	34-500	34-532			
RMDS	14-454	14-454	15-98	15-107	15-189	15-266	15-278	15-351	15-363	34-178#	34-314	34-842	34-868	34-940	
	34-:23														
RMDT	34-184#	34-284													
RMEC1	34-191#														
RMEC2	34-192#	34-<11													
RMER1	33-252	33-282	33-284	33-322	33-324	33-368	33-396	33-420	33-519	34-179#	34-320	34-852	34-941	34-:69	
RMER2	33-254	33-257	33-286	33-289	33-326	33-329	33-370	33-398	33-422	33-427	33-521	34-190#	34-942		
RMERRS	34-23#	34-216	34-805	34-810	34-820	34-940*	34-941*	34-942*	34-943*	41-4	41-4	41-4	41-4	41-7	
	41-7	41-7	41-7												
RMHR	34-188#														
RMINIT	11-95	34-212#													
RMLA	14-344	14-457	34-181#												

RMMR1	34-183#													
RMMR2	34-189#	34-943												
RMOF	34-186#	34-311	34-538	34-542	34-562	34-566								
RMR	4-144#													
RMSN	34-185#													
RMTMR	33-121	34-986#												
RMVEC	14-454*	14-454*	15-92*	15-123*	15-182*	15-213*	15-259*	15-295*	15-344*	15-382*	33-602*	34-163#	34-229	34-231
	34-346													
RMWC	14-476*	34-174#												
ROTATE	7-0	7-0	7-0	7-0	37-42#									
RPT	9-6#	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233
	15-318	16-20	16-77	16-156	16-287	16-372	33->62	33->81*	33->89	33->03*	33->53*	33->58*		
RSTART	13-5#	13-28	17-6											
RSTR1	12-72	13-13#	13-19											
RTC	4-280#													
RTURN	17-1	17-3#												
SAVCSW	7-0#	12-5	12-9*	33-;65*										
SAVEFG	11-96*	34-117#	34-591	34-718	34-928									
SAVREG	15-100	15-109	15-119	15-191	15-209	15-268	15-280	15-291	15-353	15-365	15-378	19-12	26-1#	28-1
	33-632	33-839	33-858	33-882	33-945	33-964	33-<21	33->11	33->06	33->72	34-212	34-348	34-408	34-640
	34-692	34-989	34-;93	34-<94										
	34-697	34-765#												
SC	4-230#													
SC0	4-231#													
SC1														
SC11	34-840	34-848	34-919#											
SC12	34-801	34-814	34-939#											
SC13	34-791	34-794	34-950#											
SC2	4-232#													
SC3	34-780#	34-785												
SC4	34-783#	34-817	34-831	34-833	34-884	34-905	34-917	34-937	34-979					
SC5	34-781	34-789#												
SC6	34-813	34-835#												
SC6A	34-811	34-861#												
SC7	34-850	34-864	34-866	34-876	34-887#									
SC8	34-828	34-843	34-853	34-859	34-869	34-904	34-907#	34-948						
SCOPE	4-71#	14-70	14-96	14-132	14-160	14-226	14-263	14-305	14-368	14-530	14-566	15-143	15-217	15-299
	15-386	16-63	16-134	16-260	16-363	16-389								
SCTRW	4-296#	16-49	16-89	16-313										
SEARCH	4-283#	15-95	15-103											
SEC.DS	7-0#	33-250*	33-280*	33-320*	33-366*	33-394*	33-418*	33-455*	33-576*	33-616*	33-995*	41-10	41-11	41-12
	41-23	41-26												
SEC.RD	7-0#	33-575*	41-11											
SEEK	4-275#	14-319	14-319	14-395	14-395	14-454	15-60	15-60	15-175	15-175	15-185	15-254	15-254	15-262
	15-274	15-339	15-339	15-347	15-359	16-380	16-380	16-385	16-385	33-231	33-232			
SEEKFG	12-34*	12-37*	34-125#	34-217	34-442									
SEKCN	7-0#	14-489	14-491*	14-528*										
SEKTM	7-0#	14-454*	14-528*											
SELDV	4-292#													
SERIAL	13-50	37-34#												
SET.IE	34-246	34-277	34-383	34-456	34-685	34-773	34-<36#							
SETBUF	14-409	16-231	33-945#											
SETFOR	4-291#	13-42	13-44	14-46	14-319	14-395	15-60	15-175	15-254	15-339	16-380	16-385		
SETVEC	11-94#													
SHUT	23-1	23-1	23-3#											
SKI	4-261#	33-257	33-289	33-329	33-427									
SLASH	33-;88	33->63	37-19#											
SP10	7-0#	15-216												

SP11	7-0#	15-298															
SP12	7-0#	15-385															
SP7A	7-0#	15-82	15-138														
SP7A1	7-0#	15-76															
SP7B	7-0#	15-73	15-142														
SP7B1	7-0#	15-68															
SPTYP	15-138	15-141	15-215	15-297	15-384	33-689#											
SRCHOU	15-37	15-162	15-241	15-326	33-600#												
SRCHWT	34-87#	34-410*	34-515*	34-923	34-932*												
SRTDRV	11-102	12-3#															
SRTINT	11-83	11-86#															
SRVCLK	33-105	33-115	33-120#														
ST.CLK	11-9c	15-122	15-212	15-294	15-381	17-5	33-72#										
ST.LCL	33-87	33-112#															
ST.PCL	33-81	33-102#															
STACK	4-71#	11-33	14-70	14-93	14-119	14-126	14-151	14-154	14-185	14-206	14-250	14-286	14-326	14-340			
	14-342	14-454	14-455	14-556	15-88	15-176	15-255	15-340	16-27	16-34	16-38	16-40	16-47	16-53			
	16-84	16-92	16-102	16-120	16-126	16-220	16-234	16-242	16-251	16-303	16-319	16-328	16-334	16-345			
	16-351	16-361	16-378	17-3													
STALL	14-190	14-196	14-211	14-217	16-382	16-387	33-302	33-342	33-437	33-543#							
STALL0	7-0#	14-184*	14-191	14-197	14-198*	14-199	14-201*	14-212	14-218	14-219*	14-220	14-222*					
STALL1	7-0#	33-303	33-343														
STALL2	7-0#	33-438															
STALL3	7-0#	16-383	16-388														
START	11-20	11-28#															
START1	5-1	11-11	11-18#														
START2	5-3	11-25#	12-48	13-11	23-5												
START3	5-5	11-15#															
START4	5-7	11-22#															
STATBL	7-0#	19-50															
STKLMT	4-71#																
STO	34-996	34--:16#															
STO1	34--:30#																
STO2	34--:29	34--:31	34--:33	34--:50#													
STO3	34--:52	34--:59#															
STO5	34--:66#																
STO6	34--:54	34--:69#															
STO7	34--:56	34--:78#															
STO8	34--:76	34--:84#															
STO9	34--:27	34--:37	34--:48	34--:57	34--:62	34--:64	34--:67	34--:74	34--:82	34--:86#							
STR1A	11-16	11-19#															
STR12A	11-23	11-26#															
STR1MR	15-90	15-180	15-257	15-342	33-632#												
SVADR	7-0#	16-321*	16-322*	16-340	16-341	16-357	16-358										
SVRH70	14-454	14-528	15-100	15-109	15-119	15-191	15-209	15-268	15-280	15-291	15-353	15-365	15-378	34-593			
	34-720	34-736	34-800	34-855	34-874	34-930	34-966	34--:40	34--:85	34--:93#							
SVSTAT	7-0#	19-47	33-497*														
SW0	4-71#																
SW00	4-71	4-71#	16-247														
SW01	4-71	4-71#	16-254	33-931	33--:05	33--:68											
SW02	4-71	4-71#	16-249														
SW03	4-71	4-71#	16-229	16-240													
SW04	4-71	4-71#	16-229	16-232	33-383												
SW05	4-71	4-71#	33-103	33-113	33-935												
SW06	4-71	4-71#	15-64	15-124	33-93	33-690	33-730										
SW07	4-71	4-71#	33-226	33--:57													
SW08	4-71	4-71#	12-35	18-1													

SW09	4-71	4-71#	33-929	33-:64										
SW1	4-71#													
SW10	4-71#													
SW11	4-71#													
SW12	4-71#	14-45	14-80	14-110	14-142	14-171	14-182	14-240	14-275	14-317	14-393	14-544	15-30	15-155
	15-234	15-319	16-21	16-78	16-157	16-288	16-373							
SW13	4-71#	33-544												
SW14	4-71#	33-300	33-340	33-435										
SW15	4-71#	16-299												
SW2	4-71#													
SW3	4-71#													
SW4	4-71#													
SW5	4-71#													
SW6	4-71#													
SW7	4-71#													
SW8	4-71#													
SW9	4-71#	33-468	33-585											
SWR	6-0#	11-33	11-33	11-33*	11-33*	11-33*	11-38	14-45	14-80	14-110	14-142	14-171	14-240	14-275
	14-317	14-393	14-544	15-30	15-155	15-234	15-319	16-21	16-78	16-157	16-227	16-232	16-240	16-247
	16-249	16-254	16-288	16-373	18-1	18-1	18-1	18-1	18-1	18-1	23-1	23-1	23-1*	24-1
	24-1	24-1	24-1	24-1*	24-1*	24-1*	24-1*	33-383	33-468	33-585	33-690	33-730	33-929	33-931
	33-935	33-:05	33-:64	33-:68	33-:57									
SWREG	5-1#	11-33	11-38	23-1	23-1	23-1								
T10	7-0#	15-175	15-214											
T11	7-0#	15-254	15-296											
T12	7-0#	15-339	15-383											
T7A	7-0#	15-80	15-81	15-138										
T7A1	7-0#	15-75	15-77											
T7B	7-0#	15-71	15-72	15-140										
T7B1	7-0#	15-66	15-67											
TAB.XY	5-13#	6-0												
TAP	4-224#													
TBITVE	4-71#													
TD	34-696	34-704#												
TEST0	14-44	14-70	14-70#											
TEST1	14-79	14-92	14-92#											
TEST10	14-392	14-454#	14-528	14-528										
TEST11	14-543	14-555#												
TEST12	15-29	15-87#												
TEST13	15-154	15-175#												
TEST14	15-233	15-254#												
TEST15	15-318	15-339#												
TEST16	16-20	16-26	16-26#											
TEST17	16-77	16-83	16-83#											
TEST2	14-109	14-117#												
TEST20	16-156	16-220#												
TEST21	16-287	16-318	16-318	16-318#										
TEST22	16-372	16-377	16-377#											
TEST3	14-141	14-149#												
TEST4	14-170	14-182	14-182#											
TEST5	14-239	14-247#												
TEST6	14-274	14-282#												
TEST7	14-316	14-325	14-325#											
TICKMS	7-0#	33-95*	33-98*	33-120										
TICKUS	7-0#	33-96*	33-99*											
TIM.DN	7-0#	15-369*	15-370*	33-639*	33-656									
TIM.PT	7-0#	33-635	33-674	33-676*	33-677*									

TIM.UP	7-0#	33-633	33-638*	33-653	33-736									
TIMER	14-484*	34-130#	34-451*	34-597*	34-661*	34-707*	34-836*	34-882*	34-934*	34-951*	34-975*	34-992	34-994*	34-:63*
	34-:71*	34-:78*												
TKVEC	4-71#	23-1*	23-1*											
TP50	15-67*	15-72*	15-140#											
TP60	15-75*	15-80*	15-138#											
TPB	7-0#	11-45	18-1											
TPS	7-0#	11-44	18-1											
TPS50	15-68*	15-73*	15-142#											
TPS60	15-76*	15-82*	15-138#											
TPVEC	4-71#													
TRAPVE	4-71#	11-33*	11-33*											
TRCKWC	7-0#	16-32	16-55	16-296	33-148*	33-152*								
TRE	4-83#													
TRK.DS	7-0#	33-250*	33-280*	33-320*	33-366*	33-394*	33-418*	33-455*	33-577*	33-616*	33-921*	33-994*	41-10	41-11
	41-12	41-15	41-23	41-26										
TRK.RD	7-0#	33-574*	41-11											
TRNSWT	14-480*	34-79#	34-473*	34-631	34-636*	34-647	34-667*	34-678*	34-709	34-710*	34-:34	34-:46*		
TRTVEC	4-71#													
TST0	14-44#													
TST1	14-44	14-79#												
TST10	14-316	14-392#												
TST10A	14-454	14-454#												
TST10B	14-454	14-528#												
TST11	14-392	14-543#												
TST12	14-543	15-29#												
TST13	15-29	15-154#												
TST14	15-154	15-233#												
TST15	15-233	15-318#												
TST16	15-318	16-20#												
TST17	16-20	16-77#												
TST2	14-79	14-109#												
TST20	16-77	16-156#												
TST21	16-156	16-287#												
TST22	16-287	16-372#												
TST3	14-109	14-141#												
TST4	14-141	14-170#												
TST5	14-170	14-239#												
TST6	14-239	14-274#												
TST7	14-274	14-316#												
TSTNMS	7-0#	12-12	12-14	12-69	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543
	15-29	15-154	15-233	15-318	16-20	16-77	16-156	16-287	16-372	33-131*	33-132*	33-<32*	33-<33*	33-<48*
	33-<49*	33-<55*	33-<59*	33-<63*	33-<67*	33-<71*	33-<96*	33-=30*	33-=41	33-=43				
TYPDS	19-86	26-1#												
TYPE	11-6	11-38	11-59	11-65	11-70	11-76	11-77	11-104	11-105	11-107	11-114	11-117	11-120	11-123
	11-130	11-132	11-133	11-144	11-150	12-16	12-39	12-44	12-45	12-61	12-65	12-71	13-10	13-46
	13-47	13-49	13-50	13-65	13-68	14-45	14-80	14-110	14-142	14-171	14-240	14-275	14-317	14-393
	14-544	15-30	15-155	15-234	15-319	16-21	16-78	16-157	16-288	16-373	17-1	17-1	17-1	17-1
	17-1	18-1	18-1	19-42	19-43	19-52	19-58	19-59	19-63	19-66	19-69	19-70	19-79	19-89
	19-93	19-96	19-98	19-100	19-103	20-1	21-1	22-1	23-1	23-1	23-1	23-1	23-1	23-1
	23-1	23-1	23-1	23-1	23-1	23-1	23-1	23-1	23-1	23-1	23-1	23-1	26-1#	29-1
	33-15	33-16	33-18	33-19	33-20	33-692	33-693	33-695	33-699	33-702	33-703	33-707	33-708	33-737
	33-741	33-743	33-746	33-748	33-750	33-751	33-753	33-756	33-758	33-760	33-761	33-770	33-771	33-775
	33-777	33-783	33-:59	33-:85	33-:88	33-<23	33-<39	33->33	33->35	33->57	33->59	33->61	33->63	33->86
	33->87	33->88	33->90	33-?12	33-?14									
TYPERR	18-1	19-11#												
TYPOC	11-8	17-1	19-84	23-1	26-1#	33-:87	33->89	33-?15						

TYPON	26-1#													
TYP0S	11-62	11-73	11-106	12-58	13-48	14-45	14-80	14-110	14-142	14-171	14-240	14-275	14-317	14-393
	14-544	15-30	15-155	15-234	15-319	16-21	16-78	16-157	16-288	16-373	17-1	26-1#	33-17	33->34
TYPTIM	15-138	15-139	15-214	15-296	15-383	33-728#								
ULDFLG	34-106#	34-272*	34-355	34-370*	34-553*	34-629*	34-668*	34-777	34-892	34-894*	34-919	34-921*	34-:45*	
UNLOAD	4-274#													
UNS	4-156#													
UNSTAT	11-104	37-20#												
UNTOFF	11-120	37-21#												
UNTON	11-132	37-22#												
UPE	4-107#													
US1	4-94#													
US2	4-95#													
US4	4-96#													
VERIFY	33-297	33-337	33-566#											
VV	4-129#													
WC	4-166#													
WCE	4-108#													
WCEFLG	7-0#	16-220*	16-245	33-356*	33-416*									
WCF	4-147#													
WCHKX	34-721	34-741												
WD	4-164#													
WLE	4-153#													
WRCKD	4-284#	16-37	16-54	16-104	16-127	16-243	16-342	16-359	33-379	33-403				
WRCKHD	4-285#													
WRITE	4-286#	14-396	16-35	16-48	16-86	16-121	16-235	16-298	16-323	33-385				
WRL	4-134#													
WRT.AD	34-:50*	34-:60*	34-:63#	41-6										
WRT.R1	34-:61#	34-:81												
WRT.R2	34-:47*	34-:80#												
WRT.R3	34-:56	34-:67	34-:70	34-:76	34-:82#									
WRT.R4	34-:72	34-:84#												
WRT.R5	34-:83	34-:85#												
WRT.RM	34-306	34-310	34-482	34-486	34-490	34-499	34-508	34-512	34-524	34-531	34-541	34-555	34-565	34-586
	34-603	34-624	34-826	34-857	34-902	34-:47#								
WRT.WD	34-:48*	34-:52	34-:59*	34-:62#	41-6									
WRTHD	4-287#													
XXDP	7-0#	11-50*	11-53*	11-54	11-56*	11-61	11-72	11-126	11-128	12-25	12-27			

SSCMPE	5-219#	6-0	6-0	6-0	6-0	6-0	6-0							
SSCMTM	5-219#	6-0	6-0	6-0										
SSCESA	4-71#													
SSNEWT	4-71#	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233
	15-318	16-20	16-77	16-156	16-287	16-372								
SSSET	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1#		
SSSETM	11-33	11-33#												
SSSKIP	4-71#													
.SACT1	4-32#	5-9												
.SAPT8	4-33#	6-0	6-0#											
.SAPTH	4-33#	5-12												
.SAPTY	4-33#	32-1												
.SCATC	4-30#	5-1												
.SCMTA	4-30#	5-219												
.SDB2D	4-32#	28-1												
.SDIV	4-31#	31-1												
.SEOP	4-30#	17-1												
.SERRO	4-30#	18-1												
.SRAND	4-32#	30-1												
.SRDE	4-31#													
.SRDOC	4-31#													
.SREAD	4-31#	23-1												
.SSAVE	4-32#	25-1												
.SSB2D	4-32#	27-1												
.SSCOP	4-32#	24-1												
.SSIZE	4-32#													
.SSUPR	4-32#	29-1												
.STRAP	4-30#	26-1												
.STYPD	4-31#	22-1												
.STYPE	4-30#	20-1												
.STYPO	4-31#	21-1												
.EQUAT	4-30#	4-71												
.HEADE	4-31#	4-37												
.SETUP	4-31#	4-298												
.SWRHI	4-30#	4-38												
.SWRLO	4-38#	4-39	4-40	4-41	4-42	4-43	4-44	4-45	4-46					
CKCHR	10-409#	33-;91	33-<26	33->66	33->93	33-?18	33-a79	33-a86	33-a91	33-A36				
CKDIG	10-419#	33-<28	33->80											
CKNUM	10-429#	33-;93	33-?02	33-?20										
COMMEN	4-71#													
COMIND	10-346#	13-44	14-46	14-319	14-395	15-60	15-175	15-254	15-339	16-380	16-385			
DO	10-355#	13-43	13-45	14-70	14-70	14-94	14-95	14-121	14-128	14-152	14-156	14-187	14-193	14-208
	14-214	14-252	14-254	14-292	14-294	14-296	14-298	14-300	14-302	14-341	14-361	14-557	14-558	15-60
	15-175	15-254	15-339	16-381	16-386	33-583								
DODTA	10-370#	16-36	16-39	16-43	16-52	16-58	16-95	16-106	16					

FRRTP	10-393#	18-1												
ESCAPE	4-71#	15-60	15-175	15-254	15-339	33-249	33-279	33-319	33-365	33-393	33-417	33-454	33-579	33-615
	33-922	33-927	33-997	33-:03	33-:50									
EXIT	4-8#	14-70	14-96	14-132	14-160	14-226	14-263	14-305	14-368	14-530	14-566	15-143	15-217	15-299
	15-386	16-63	16-134	16-260	16-363	16-389								
GETPRI	4-71#	31-1												
GETSWR	4-71#	11-38	11-38#											
LOOP	10-359#	14-119	14-126	14-151	14-154	14-185	14-206	14-250	14-286	14-340	14-342	14-454	14-455	16-34
	16-38	16-40	16-47	16-53	16-92	16-102	16-120	16-126	16-234	16-242	16-251	16-303	16-328	16-334
	16-345	16-351	16-361											
MORE.S	10-326#	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233
	15-318	16-20	16-77	16-156	16-287	16-372								
MORE.TA	5-15#	6-0	8-1#											
MSG	14-37#	14-44	14-72#	14-79	14-98#	14-109	14-134#	14-141	14-162#	14-170	14-228#	14-239	14-265#	14-274
	14-307#	14-316	14-370#	14-392	14-532#	14-543	15-13#	15-29	15-145#	15-154	15-219#	15-233	15-301#	15-318
	16-9#	16-20	16-65#	16-77	16-136#	16-156	16-262#	16-287	16-365#	16-372				
MULT	4-71#													
NEWTST	4-71#	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233
	15-318	16-20	16-77	16-156	16-287	16-372								
POP	4-71#	22-1	25-1	30-1	31-1	32-1	32-1	33-171	33-216	33-784				
PUSH	4-71#	22-1	25-1	30-1	31-1	32-1	32-1	32-1	33-130	33-180	33-728			
REPORT	4-71#	10-296#	15-214	15-296	15-383									
SAV.RH	10-280#	15-100	15-109	15-119	15-191	15-209	15-268	15-280	15-291	15-353	15-365	15-378		
SET.TN	10-301#	14-44	14-79	14-109	14-141	14-170	14-239	14-274	14-316	14-392	14-543	15-29	15-154	15-233
	15-318	16-20	16-77	16-156	16-287	16-372								
SETPRI	4-71#	23-1												
SETTRA	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1	26-1#		
SETUP	4-71#	11-33												
SKIP	4-71#													
SLASH	4-71#	14-1	14-25	14-29	14-35	15-3	15-11	16-5	16-7					
STARS	4-71#	4-270	5-9	5-12	5-12	5-12	6-0	6-0	6-0	8-227	8-245	14-44	14-44	14-79
	14-79	14-109	14-109	14-141	14-141	14-170	14-170	14-239	14-239	14-274	14-274	14-316	14-316	14-392
	14-392	14-543	14-543	15-29	15-29	15-154	15-154	15-233	15-233	15-318	15-318	16-20	16-20	