

RM02/03/05

RM05/3/2 FCTNL TST 2
CZRMNBO

AH-F925B-MC
FICHE 1 OF 2

AUG 1981
COPYRIGHTS 80-81
MADE IN USA



The main body of the document is a large, dense grid of data. Each cell in the grid contains a small, structured table or form. The text within these cells is extremely faint and difficult to read, but the overall layout suggests a comprehensive data set or a series of related reports. The grid covers most of the page area below the header.

RM02/03/05

RM05/3/2 FCTNL TST 2
CZRMNBO

AH-F925B-MC
FICHE 2 OF 2

AUG 1981
COPYRIGHT © 80-81
MADE IN USA



.REM \

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

IDENTIFICATION

PRODUCT CODE: AC-F924B-MC
PRODUCT NAME: CZRMNBO RM05/3/2 FUNCTIONAL TEST, PT 2
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
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

- 1. INTRODUCTION
 - 1. ABSTRACT
 - 2. UNIT UNDER TEST
- 2. OPERATING REQUIREMENTS
 - 1. HARDWARE REQUIREMENTS
 - 2. MEDIA REQUIREMENTS
 - 3. PREREQUISITE DIAGNOSTIC PROGRAMS
- 3. OPERATING PROCEDURE
 - 1. LOADING
 - 2. SWITCH OPTIONS
 - 3. STARTING
 - 4. HALTING
 - 5. RESTARTING
- 4. OPERATOR INTERFACE
 - 1. PROGRAM I.D.
 - 2. CONSOLE DIALOGUE
 - 3. PROGRESS REPORTS
 - 4. PERFORMANCE REPORTS
 - 5. PROGRAM HALTS
 - 6. ERROR REPORTS
 - 7. EXECUTION TIME
- 5. ENVIRONMENTAL SUPPORT
 - 1. PROCESSOR COMPATIBILITY
 - 2. DUAL PORT CONFIGURATIONS
 - 3. MEMORY PARITY HARDWARE
 - 4. MEMORY MANAGEMENT HARDWARE
 - 5. ACT, APT COMPATIBILITY
 - 6. XXDP COMPATIBILITY
 - 7. OPERATING SYSTEM COMPATIBILITY
- 6. TEST DESCRIPTION

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

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL MEANS TO VERIFY THE OPERABILITY OF THE RM05/3/2 DISK SUBSYSTEM. IN PARTICULAR, THE PROGRAM SERVES THE FOLLOWING PURPOSES:

TO EXPLICITLY ESTABLISH CONFIDENCE IN THE BASIC OPERATIONS OF THE DISK DRIVE, INCLUDING MECHANICAL POSITIONING AND DATA TRANSFER OPERATIONS;

TO IMPLICITLY ESTABLISH CONFIDENCE IN THE DRIVE/ADAPTER ELECTRICAL INTERFACE;

TO VERIFY THE FUNCTIONALITY OF THE RM05/3/2 SUBSYSTEM, INCLUDING THE MASSBUS CONTROLLER, MASSBUS ADAPTER AND THE DISK DRIVE.

THE TEST IS COMPRISED OF 3 PARTS, WHICH WOULD NORMALLY BE RUN IN SEQUENCE, STARTING WITH PART 1. BRIEFLY, PART 1 TESTS HOUSEKEEPING AND MECHANICAL POSITIONING OPERATIONS; PART 2 TESTS WRITE, READ AND WRITE CHECK OPERATIONS USING HEADER AND DATA; PART 3 TESTS WRITE, READ AND WRITE CHECK OPERATIONS USING DATA.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM05/3/2 DISK SUBSYSTEM WHICH CONSISTS OF THE RH MASSBUS CONTROLLER, THE RM05/3/2 MASSBUS ADAPTER, AND THE STORAGE MODULE DISK DRIVE. NOTE THAT A DISK PACK IS REQUIRED FOR TESTING AND IS CONSIDERED AN INTEGRAL OF THE STORAGE MODULE DISK DRIVE.

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

THE FOLLOWING MINIMUM HARDWARE CONFIGURATION, ASSUMED TO BE OPERATIONAL, IS REQUIRED TO LOAD AND EXECUTE THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST:

PDP-11 PROCESSOR
20K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114

2.2 MEDIA REQUIREMENTS

EACH UNIT BEING TESTED MUST BE LOADED WITH A SCRATCH DISK PACK BEFORE TESTING BEGINS ON THAT UNIT. THE DISK MUST BE FORMATTED AND CONTAIN A READABLE COPY OF THE MFG AND USR BAD SECTOR FILES.

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

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

RM05/3/2 FUNCTIONAL TEST, PART 1

3.0 OPERATING PROCEDURE

3.1 LOADING

THE PROGRAM MAY BE LOADED BY EITHER OF THE FOLLOWING MEDIA:

- .PAPER TAPE, USING THE STANDARD PAPER TAPE LOADING PROCEDURE.
- .XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

3.2 SWITCH OPTIONS

THE FOLLOWING SWITCH OPTIONS ARE PROVIDED TO ENHANCE THE UTILITY OF THE PROGRAM.

- SW15 HALT ON ERROR
- SW14 LOOP ON TEST (CURRENTLY BEING EXECUTED)
- SW13 INHIBIT ERROR TYPEOUTS
- SW12 UNUSED
- SW11 INHIBIT TEST ITERATIONS
- SW10 BELL ON ERROR
- SW09 LOOP ON ERROR
- SW08 LOOP ON TEST IN SW07-00

THE LOW ORDER 8 SWITCHES (SW07-SW00), ARE USED IN CONJUNCTION WITH SW08 TO SPECIFY THE OCTAL NUMBER OF THE TEST WHICH THE PROGRAM WILL LOOP ON.

3.3 STARTING

THE PROGRAM MAY BE STARTED AT LOCATION 200 OR 204. STARTING AT 200 WILL BE THE NORMAL STARTING ADDRESS. STARTING AT 204 WILL ENABLE THE RH/RM BASE ADDRESS TO BE CHANGED. IF RUNNING IN A STAND-ALONE ENVIRONMENT, THE PROGRAM USES CONSOLE DIALOGUE TO ALLOW THE OPERATOR TO CONTROL TEST CONDITIONS.

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

3.4 HALTING

THE PROGRAM SHOULD BE HALTED BY TYPING CONTROL C FROM THE CONSOLE.

NOTE: IF THE PROGRAM IS HALTED BY ANY OTHER MEANS, BAD HEADER INFORMATION MAY BE LEFT ON THE DISK PACK. THIS OF COURSE DEPENDS ON WHICH TEST IS BEING PERFORMED AT THE TIME OF THE HALT.

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200 OR 204. (SEE SECTION 3.3)

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

THE PROGRAM TYPES ITS TITLE AND MAINDEC NUMBER THE FIRST TIME IT IS STARTED AFTER BEING LOADED. ALSO, A WARNING MESSAGE IS TYPED, NOTIFYING THE OPERATOR OF POSSIBLE HEADER CORRUPTION IF THE PROGRAM IS HALTED IMPROPERLY. THE PROGRAM IDENTIFICATION AND THE WARNING DO NOT OCCUR IF THE PROGRAM IS RESTARTED.

4.2 CONSOLE DIALOGUE

WHEN THE PROGRAM IS RUNNING IN STAND ALONE MODE, IT ENTERS A CONSOLE DIALOGUE SEQUENCE AFTER TYPING THE PROGRAM I.D. AND WARNING MESSAGE. (SEE SECTION 4.1)

THE FIRST QUESTION TYPED OUT IS: 'TYPE HELP TEXT (L) N ?'. IF THE OPERATOR RESPONDS WITH A 'Y', THE PROGRAM WILL TYPE A BRIEF HELP MESSAGE WHICH WILL LIST SWITCH OPTIONS, ETC. ANY OTHER RESPONSE TO THE QUESTION IS CONSIDERED A 'N' AND NO HELP TEXT IS TYPED. THIS QUESTION IS ONLY ASKED ON THE INITIAL PROGRAM START AND NOT ON SUBSEQUENT START-UP'S.

ON THE PROGRAM INITIAL START AND WHEN RESTARTING AT LOCATION 204, THE OPERATOR MAY CHANGE THE RH/RM BASE ADDRESSES WITH THE FOLLOWING DIALOGUE.

EXAMPLE 1

```
RMCS1=176700 <CR>      ;NO CHANGE IN ADDRESS
RMVEC=000254 <CR>      ;NO CHANGE IN ADDRESS
```

EXAMPLE 2

```
RMCS1=176700 177200<CR> ;CHANGE BASE ADDRESS TO 177200
RMVEC 000254 260<CR>   ;CHANGE VECTOR ADDRESS TO 260
```

ON THE INITIAL START, THE NEXT QUESTION TYPED IS, 'TYPE 'A' 'C

172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228

TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S) AND TERMINATE INPUT WITH A CARRIAGE RETURN'. THEN, 'DRIVE(S):' IS TYPED AND WAITS FOR THE OPERATOR TO TYPE AN 'A', TO TEST ALL POSSIBLE DRIVES OR TYPE ANY STRING OF DRIVE NUMBER(S) TO BE TESTED AND TERMINATE THE INPUT WITH A 'CARRIAGE RETURN'. NO COMMAS OR ANY OTHER SEPARATORS ARE NEEDED WHEN ENTERING THE DRIVE NUMBERS AS A STRING. THE PROGRAM ENTERS THE COMMA SEPARTOR AUTOMATICALLY AFTER TYPING EACH NUMBER. ON ALL SUBSEQUENT STARTS, ONLY THE 'DRIVE(S):' PROMPT IS TYPED.

THE DIAGNOSTIC THEN INITIALIZES AND REPORTS THE STATUS OF THE DRIVES WHICH WERE PREVIOUSLY SPECIFIED FOR TESTING. THE FOLLOWING IS AN EXAMPLE PRINTOUT:

```
'UNIT STATUS:
0 ONLINE RM03
1 LOAD DEVICE
2 OFFLINE RM05
3 NOT PRESENT
4 NOT PRESENT
5 NOT AN RM05/3/2
6 NOT PRESENT
7 NOT PRESENT'
```

THE ABOVE UNIT STATUS SHOWS THAT DRIVE 0 WILL BE TESTED, WHILE DRIVES 1 - 7 WILL NOT BE TESTED.

THE DIAGNOSTIC THEN TYPES THE FOLLOWING MESSAGE, BASED ON THE STATUS OF THE DRIVE:

'DRIVE(S) TO BE TESTED, 0'

IF NO DRIVES ARE AVAILABLE FOR TESTING, THE FOLLOWING MESSAGE WILL BE TYPED TO THE OPERATOR:

'DRIVE(S) TO BE TESTED, NONE'

THE PROGRAM WILL THEN, EITHER START TESTING THE DRIVES AVAILABLE FOR TESTING OR RETURN TO THE BEGINNING OF THE PROGRAM AND WAIT.

ONCE THE DRIVES START TESTING, THE FOLLOWING MESSAGE WILL OCCUR AS EACH DRIVE BEGINS TO BE TESTED:

'DRIVE 0'

AFTER ALL THE DRIVES ARE COMPLETELY TESTED, THE END OF PASS MESSAGE WILL BE TYPED (SEE SECTION 4.3) AND THE PROGRAM WILL START TESTING ALL THE DRIVES AGAIN. THIS WILL CONTINUE UNTIL THE PROGRAM IS HALTED BY THE OPERATOR.

NOTE: THE LETTER LOCATED WITHIN THE BRACKETS () INDICATES THE TYPE OF RESPONSE REQUIRED BY THE USER, D DECIMAL, O-OCTAL AND L=LETTER.

4.3 PROGRESS REPORTS

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

AN END OF PASS REPORT OCCURS EACH TIME THE PROGRAM IS EXECUTED FOR ALL DEVICES IN THE TEST QUE. THE END OF PASS REPORT IS AS FOLLOWS.

'END OF PASS 1'

THE FOLLOWING MESSAGE WILL ALSO OCCUR IF THERE WERE ERRORS SINCE THE LAST END OF PASS REPORT.

'TOTAL ERRORS SINCE LAST REPORT 0'

4.4 PERFORMANCE REPORT

NO PERFORMANCE REPORTS ARE GIVEN DURING THE EXECUTION OF THE PROGRAM.

4.5 PROGRAM HALTS

THERE ARE NO SCHEDULED HALTS DURING THE EXECUTION OF THE PROGRAM. PROCESSOR HALTS ARE DUE TO THE TRAP CATCHER.

4.6 ERROR REPORTS

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE NUMBER OF THE UNIT (DRIVE) BEING TESTED, THE DRIVE TYPE, THE TEST NUMBER, THE ERROR NUMBER AND THE VALUE OF THE PROGRAM COUNTER WHERE THE ERROR WAS CALLED. THIS LINE IS FOLLOWED BY THE ERROR MESSAGE: ONE OR MORE LINES OF TEXT WHICH GIVE A BRIEF, YET COMPREHENSIVE DESCRIPTION OF THE ERROR. THE ERROR MESSAGE IS NORMALLY FOLLOWED BY ONE OR MORE PAIRS OF LINES CONTAINING DATA HEADERS AND DATA PERTINENT TO THE ERROR, INCLUDING EXPECTED AND ACTUAL TEST RESULTS.

THE FOLLOWING PRINTOUT SHOWS A TYPICAL ERROR MESSAGE FOR THIS PROGRAM:

```

DRV# 0 - RM03, TEST# 14, ERR# 326, PC=016654
MASSBUS DATA BUS PARITY ERROR 'MDPE' (RMCS2, BIT 8) DETECTED
DURING WRITE COMMAND
EXPECTED      RECEVD
040300      040700
RMCS1      RMCS2      RMDS      RMER1      RMER2      RMAS
144252      040700      010700      000000      000000      000000
RMWC      RMBA      RMDA      RMOF      RMDC      RMEC1      RMEC2
177403      104604      000002      010000      000000      004066      000000
RMMR1      RMMR2      RMDT      RMSN
000010      011777      024026      177777

```

4.7 EXECUTION TIME

PASS 1 OF THE PROGRAM TAKES ABOUT 6 SECONDS. PASS 2 AND SUBSEQUENT PASSES TAKE 35 SECONDS.

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

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET, AND PROVIDING THAT DATA THROUGHPUT ON THE SYSTEM IS SUFFICIENT TO SUSTAIN DATA TRANSFER OPERATIONS.

5.2 DUAL PORT CONFIGURATIONS

THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST DOES NOT SPECIFICALLY TEST DUAL PORT LOGIC IN THE RM05/3/2 ADAPTER BUT IS EXECUTABLE ON RM05/3/2 SUBSYSTEMS HAVING THE DUAL PORT OPTION PROVIDING THE DUAL PORT SWITCH IS SET TO THE APPROPRIATE PORT (A OR B).

5.3 MEMORY PARITY HARDWARE

MEMORY PARITY HARDWARE IS NOT USED DURING THE EXECUTION OF THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE IS NOT USED DURING THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST. CAPABILITIES OF THE MASSBUS CONTROLLER.

5.5 ACT11, APT11 COMPATIBILITY

THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH ACT11 AND APT11 IN BOTH DUMP AND AUTOMATIC MODES. FURTHER, THE PROGRAM WILL EXECUTE A QUICK PASS DURING THE FIRST PASS IN SUPPORT OF QUICK VERIFY MODE.

5.6 XXDP COMPATIBILITY

THE RM05/3/2 SUBSYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES, AND PROVIDES MEDIA PROTECTION IN THE CASE WHERE THE RM05/3/2 IS THE XXDP LOADING DEVICE.

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399

6.0 TEST DESCRIPTION

TEST 1 CONTROLLER ACCESS TEST

PURPOSE:

TO VERIFY THAT THE UNIBUS ADDRESS OF THE SUBSYSTEM IS CORRECT, AS DEFINED AT LOCATION \$BASE.

PROCEDURE:

THE TEST TRIES TO ACCESS ALL MASSBUS CONTROLLER REGISTERS USING THE \$BASE ADDRESS. REGISTER CONTENTS ARE IGNORED DURING THE TEST, AND THE TEST FAILS IF A BUS TIMEOUT OCCURS FOR ANY REGISTER TRANSFER.

IF THE TEST FAILS AND THE PROGRAM IS RUNNING IN A STAND ALONE ENVIRONMENT, I.E., LOCATION 42 IS 0, THE PROGRAM WILL JUMP TO LOCATION 204 WHICH ALLOWS THE OPERATOR TO CHANGE THE \$BASE ADDRESS VIA CONSOLE DIALOGUE. OTHERWISE, THE PROGRAM ESCAPES TO THE END OF PASS HANDLER.

TEST 2 - 31 WRITE/READ HEADER AND DATA (FORMAT) TESTS

PURPOSE:

TO TEST WRITE HEADER AND DATA AND READ HEADER AND DATA FUNCTIONALITY OF THE RM05/3/2 SUBSYSTEM USING A SET OF VARIABLES WHICH INCLUDE WORD COUNT, HEAD MOTION, HEAD SWITCHING AND ERROR CONDITIONS.

PROCEDURE:

ALTHOUGH EACH TEST EXERCISES A DIFFERENT VARIABLE, THE GENERAL PROCEDURE OF EACH TEST IS THE SAME. THE DRIVE IS INITIALIZED AND RECALIBRATED IF 'PIP' OR 'SKI' ARE ACTIVE SO THAT THERE ARE NO ERRORS WHEN A TEST BEGINS. FOLLOWING THAT, THE TEST PERFORMS ANY EXPLICIT SEEKS REQUIRED FOR THE CONDITIONS OF THE TEST. REGISTERS ARE PRESET AND THE WRITE HEADER AND DATA COMMAND IS EXECUTED. WHEN THE WRITE COMMAND IS COMPLETE, THE TEST STORES ALL SUBSYSTEM STATUS AND CHECKS FOR PRIMARY ERRORS WHICH PRECLUDE OTHER STATUS CHECKS. IF THERE ARE NO PRIMARY ERRORS, THE TEST VERIFIES THE RESULTS OF THE WRITE COMMAND AND THEN CHECKS FOR SECONDARY ERRORS. LOOP ADDRESSES ARE MODIFIED FOLLOWING THE SUCCESSFUL COMPLETION OF THE WRITE COMMAND IN ORDER TO SHORTEN EXECUTION TIMES AND ENHANCE SCOPING LOOPS, THEN THE PROGRAM EXECUTES THE READ HEADER AND DATA PORTION OF THE TEST, VERIFYING THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.

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

NOTE: THE SECTOR USED DURING A TEST MAY DIFFER FROM THE PROGRAM LISTING BECAUSE THE PROGRAM SUBSTITUTES A GOOD SECTOR, IF THE ONE SELECTED IS LISTED IN THE BAD BLOCK TABLE.

TEST 2 FORMAT ZEROS TEST

THE TEST SEEKS TO CYLINDER 0, THEN WRITES HEADER AND DATA ON SECTOR 0 IN 18 BIT FORMAT. THE HEADER AND DATA FIELDS ARE ALL ZEROS, CAUSING THE DEVICE TO USE NORMAL WRITE GATE. THE HEADER AND DATA ARE READ AND COMPARED WITH THE WRITE BUFFER. THE INITIAL SEEK POSITIONS THE HEAD SUCH THAT THERE IS NO IMPLIED SEEK. THE TEST IS REPEATED FOR 16 BIT FORMAT.

TEST 3 ZERO FILL TEST TEST

THE TEST EXECUTES A SEEK TO CYLINDER 0 TO INSURE THAT THERE IS NO HEAD MOTION DURING DATA TRANSFER. THIS IS FOLLOWED BY A WRITE HEADER AND DATA COMMAND WITH THE WORD COUNT EQUAL TO THE SIZE OF THE HEADER WHICH CAUSES THE RH TO ZERO FILL THE DATA FIELD. THE READ HEADER AND DATA COMMAND THAT FOLLOWS READS A FULL SECTOR AND VERIFIES THAT DATA WAS ZERO FILLED.

TEST 4 FORMAT CHECK ZEROS TEST

THE TEST WRITES HEADER AND AN ALL ZEROS DATA FIELD, THEN PERFORMS A WRITE CHECK HEADER AND DATA COMMAND AND VERIFIES THERE ARE NO ERRORS.

TEST 5 FORMAT CHECK ZEROS W/ WCE ERROR TEST

THE TEST WRITES HEADER AND AN ALL ZEROS DATA FIELD. AFTER COMPLEMENTING THE LAST WORD OF THE WRITE BUFFER, THE TEST PERFORMS A WRITE CHECK HEADER AND DATA COMMAND AND VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

TEST 6 FORMAT ONES TEST

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

THE TEST WRITES HEADER AND AN ALL ONES DATA FIELD, THEN READS THE HEADER AND DATA, VERIFYING THE READ BUFFER WITH THE WRITE BUFFER. THE ALL ONES FIELD IS A CONSTANT FREQUENCY, AND THE DRIVE SHOULD USE NORMAL WRITE GATE.

TEST 7 FORMAT CHECK ONES TEST

THE TEST FORMATS AN ALL ONES DATA FIELD, THEN PERFORMS A WRITE CHECK HEADER AND DATA COMMAND, VERIFYING THAT THERE ARE NO ERRORS.

TEST 10 FORMAT CHECK ONES W/ WCE ERROR TEST

THE TEST FORMATS AN ALL ONES DATA FIELD, THEN COMPLEMENTS THE LAST WORD OF THE WRITE BUFFER. A WRITE CHECK HEADER AND DATA COMMAND IS EXECUTED, AND THE TEST VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

TEST 11 FORMAT MULTIPLE SECTORS TEST

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND FOLLOWS, WITH THE WORD COUNT EQUAL TO MULTIPLE SECTORS. THE SAME SECTORS ARE VERIFIED WITH A WRITE CHECK HEADER AND DATA COMMAND.

TEST 12 FORMAT WITH HEAD SWITCHING TEST

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND STARTS WITH CYLINDER 0, TRACK 0, SECTOR 31. THE WORD COUNT IS EQUAL TO MULTIPLE SECTORS WHICH CAUSES THE SUBSYSTEM TO SWITCH FROM TRACK 0 TO TRACK 1 AFTER THE FIRST OF THE MULTIPLE SECTORS ARE WRITTEN. THE SAME SECTORS ARE VERIFIED WITH A WRITE CHECK HEADER AND DATA COMMAND, USING THE SAME WORD COUNT AND STARTING SECTOR.

514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570

TEST 13 FORMAT WITH MID-TRANSFER SEEK TEST

THIS TEST WRITES MULTIPLE SECTORS STARTING WITH CYLINDER 0, LAST TRACK AND SECTOR 31., CAUSING A MIDTRANSFER SEEK AFTER THE FIRST SECTOR IS WRITTEN. THE SAME SECTORS ARE VERIFIED WITH WRITE CHECK HEADER AND DATA COMMAND.

TEST 14 FORMAT WITH IMPLIED SEEK TEST

THIS TEST SEEKS TO THE LAST CYLINDER PRIOR TO WRITING HEADER AND DATA ON CYLINDER 0. THE EXPLICIT SEEK INSURES THAT THERE WILL BE MAXIMUM HEAD MOTION DURING THE IMPLIED SEEK OF THE WRITE COMMAND. THE SAME OPERATION, INCLUDING THE EXPLICIT SEEK IS REPEATED FOR READ HEADER AND DATA.

TEST 15 FORMAT EACH SECTOR ADDRESS TEST

HEADERS AND DATA OF EACH SECTOR ON CYLINDER 0, TRACK 0 ARE FORMATTED AND READ WITH THE PROGRAM VERIFYING HEADERS AND DATA.

TEST 16 FORMAT EACH TRACK ADDRESS TEST

THIS TEST FORMATS SECTOR 0 OF EACH TRACK ON CYLINDER 0 AND READS EACH SECTOR WITH THE PROGRAM VERIFYING HEADERS AND DATA.

TEST 17 FORMAT PRIME CYLINDERS TEST

THIS TEST FORMATS AND READS SECTOR 0, TRACK 0 ON EACH PRIME CYLINDER, I.E., CYLINDERS 1, 2, 4, 8., 16., 32., 64., 128., 256., AND 512.

TEST 20 READ HEADER & DATA IN LAST SECTOR TEST

THIS TEST READS HEADER AND DATA ON THE LAST SECTOR OF THE, DISK, I.E., AND VERIFIES THAT 'LBT' STATUS SETS.

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

TEST 21 READ HEADER & DATA W/ AOE ERROR TEST

THIS TEST READS MULTIPLE SECTORS STARTING WITH THE LAST SECTOR 31. AND VERIFIES THAT AOE STATUS SETS.

TEST 22 READ INVALID SECTOR ADDRESS TEST

THIS TEST USES AN ILLEGAL SECTOR ADDRESS AND VERIFIES THAT IAE STATUS SETS.

TEST 23 READ INVALID TRACK ADDRESS TEST

THIS TEST USES AN ILLEGAL TRACK ADDRESS AND VERIFIES THAT IAE STATUS SETS.

TEST 24 READ INVALID CYLINDER ADDRESS TEST

THIS TEST USES AN ILLEGAL CYLINDER ADDRESS AND VERIFIES THAT IAE STATUS SETS.

TEST 25 FORMAT AT OFFSET TEST

THE PROGRAM SETS OFFSET MODE AND EXECUTES A WRITE HEADER AND DATA COMMAND, VERIFYING THAT OFFSET MODE IS RESET BY THE WRITE COMMAND.

TEST 26 IVC FORMAT TEST

VOLUME VALID IS RESET BY SETTING AND RESETTING DIAGNOSTIC MODE. THE TEST THEN EXECUTES A WRITE HEADER AND DATA COMMAND AND VERIFIES THAT INVALID COMMAND STATUS SETS. THE TEST IS REPEATED FOR READ HEADER AND DATA COMMAND.

628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650

TEST 27 FORMAT ERROR TEST

A SINGLE SECTOR IS FORMATTED WITH THE OFFSET REGISTER SET FOR 16 BIT FORMAT AFTER WHICH THE SAME SECTOR IS READ IN 18 BIT FORMAT WITH THE PROGRAM VERIFYING THAT FORMAT ERROR STATUS IS SET. THE SAME PROCEDURE IS REPEATED WITH THE SECTOR WRITTEN IN 18 BIT FORMAT AND READ IN 16 BIT FORMAT.

TEST 30 & 31 FORMAT HCE TEST (1ST AND 2ND HEADER WORDS)

THESE TWO TESTS WRITE AN INCORRECT HEADER THEN READ THE HEADER AND VERIFY THAT THE CORRECT HEADER ERROR IS DETECTED. THE TESTS SETUP THE CORRECT HEADER, THEN COMPLEMENT BIT 0 AND USE THE MODIFIED BUFFER TO WRITE THE HEADER. THE PROCESS IS REPEATED UNTIL EACH BIT POSITION HAS BEEN SEPARATELY TESTED.

479
480

```

;*LAST REVISION 04-APR-81
.TITLE CZRMNBO RM05/3/2 FCTNL TST 2
;*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

```

481

```

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH      USE
;*      -----
;*      15          HALT ON ERROR
;*      14          LOOP ON TEST
;*      13          INHIBIT ERROR TYPEOUTS
;*      12          UNUSED
;*      11          INHIBIT ITERATIONS
;*      10          BELL ON ERROR
;*      9           LOOP ON ERROR
;*      8           LOOP ON TEST IN SWR<7:0>
;*      7           TN128
;*      6           TN64
;*      5           TN32
;*      4           TN16
;*      3           TN8
;*      2           TN4
;*      1           TN2
;*      0           TN1

```

482

483
484

```

.SBTTL BASIC DEFINITIONS

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

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

;*GENERAL PURPOSE REGISTER DEFINITIONS
R0 = %0            ;;GENERAL REGISTER
R1 = %1            ;;GENERAL REGISTER
R2 = %2            ;;GENERAL REGISTER
R3 = %3            ;;GENERAL REGISTER

```

BASIC DEFINITIONS

```

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 RM REGISTER BIT DEFINITIONS

*RMCS1 CONTROL STATUS REGISTER

```

004000 DVA = BIT11 ;:DEVICE AVAILABLE-READ ONLY
000040 F4 = BIT05 ;:FUNCTION CODE
000020 F3 = BIT04 ;:FUNCTION CODE
000010 F2 = BIT03 ;:FUNCTION CODE
000004 F1 = BIT02 ;:FUNCTION CODE
000002 F0 = BIT01 ;:FUNCTION CODE
000001 GO = BIT00 ;:GO BIT
000077 FNCMSK = 000077 ;:FUNCTION CODE MASK
    
```

*FUNCTION CODES (BITS 01-05 OF RMCS1)

```

485
486
487
488
489
490 000000 NOP = 000000 ;:NOP COMMAND
491 000002 ILF02 = 000002 ;:ILLEGAL COMMAND
492 000004 SEEK = 000004 ;:SEEK COMMAND
493 000006 RECAL = 000006 ;:RECALIBRATE COMMAND
494 000010 DRVCLR = 000010 ;:DRIVE CLEAR COMMAND
495 000012 RELEASE = 000012 ;:RELEASE COMMAND
496 000014 OFFSET = 000014 ;:OFFSET COMMAND
497 000016 RTC = 000016 ;:RETURN TO CENTERLINE COMMAND
498 000020 RIP = 000020 ;:READ IN PRESET COMMAND
499 000022 PAKACK = 000022 ;:PACK ACKNOWLEDGE COMMAND
500 000024 ILF24 = 000024 ;:ILLEGAL COMMAND
501 000026 ILF26 = 000026 ;:ILLEGAL COMMAND
502
503
504
505
506
507
508
509
510
511
512
    
```

513	000030	SEARCH = 000030	:SEARCH COMMAND
516	000030	ILF30 = 000030	:ILLEGAL COMMAND
	000032	ILF32 = 000032	:ILLEGAL COMMAND
	000034	ILF34 = 000034	:ILLEGAL COMMAND
	000036	ILF36 = 000036	:ILLEGAL COMMAND
	000040	ILF40 = 000040	:ILLEGAL COMMAND
	000042	ILF42 = 000042	:ILLEGAL COMMAND
	000044	ILF44 = 000044	:ILLEGAL COMMAND
	000046	ILF46 = 000046	:ILLEGAL COMMAND
517	000050	WCD = 000050	:WRITE CHECK DATA COMMAND
518	000052	WCH = 000052	:WRITE CHECK HEADER AND DATA
519	000054	ILF54 = 000054	:ILLEGAL COMMAND
520	000056	ILF56 = 000056	:ILLEGAL COMMAND
521	000060	WD = 000060	:WRITE DATA COMMAND
522	000062	WH = 000062	:WRITE HEADER AND DATA COMMAND
523	000064	ILF64 = 000064	:ILLEGAL COMMAND
524	000066	ILF66 = 000066	:ILLEGAL COMMAND
525	000070	RD = 000070	:READ DATA COMMAND
526	000072	RH = 000072	:READ HEADER AND DATA COMMAND
527	000074	ILF74 = 000074	:ILLEGAL COMMAND
528	000076	ILF76 = 000076	:ILLEGAL COMMAND
529			
530		:*RMDA DISK ADDRESS REGISTER	
531			
532		:TRACK ADDRESS DEFINITIONS	
533	010000	TA16 = BIT12	:TRACK ADDRESS 16.
534	004000	TA8 = BIT11	:TRACK ADDRESS 8.
535	002000	TA4 = BIT10	:TRACK ADDRESS 4
536	001000	TA2 = BIT09	:TRACK ADDRESS 2
537	000400	TA1 = BIT08	:TRACK ADDRESS 1
538			
539		:SECTOR ADDRESS DEFINITIONS	
540	000020	SA16 = BIT04	:SECTOR ADDRESS 16.
541	000010	SA8 = BIT03	:SECTOR ADDRESS 8.
542	000004	SA4 = BIT02	:SECTOR ADDRESS 4
543	000002	SA2 = BIT01	:SECTOR ADDRESS 2
544	000001	SA1 = BIT00	:SECTOR ADDRESS 1
545			
546		:TRACK & SECTOR MASKS	
547	177400	TADMSK = 177400	:TRACK ADDRESS MASK
548	000377	SADMSK = 000377	:SECTOR ADDRESS MASK
549			
550		:*RMDS DRIVE STATUS REGISTER	
551			
552	100000	ATA = BIT15	:ATTENTION ACTIVE
553	040000	ERR = BIT14	:COMPOSITE ERROR
554	020000	PIP = BIT13	:POSITIONING IN PROGRESS
555	010000	MOL = BIT12	:MEDIUM ON LINE
556	004000	WRL = BIT11	:WRITE LOCK
557	002000	LBT = BIT10	:LAST BLOCK TRANSFERRED
558	001000	PGM = BIT09	:PROGRAMMABLE
559	000400	DFR = BIT08	:DRIVE PRESENT
560	000200	DRY = BIT07	:DRIVE READY
561	000100	VV = BIT06	:VOLUME VALID
562	000001	OM = BIT00	:OFFSET MODE ACTIVE
563			
564		:*RMER1 ERROR REGISTER #1	

```

565
566      100000      DCK      - BIT15      ;DATA CHECK ERROR
567      040000      UNS      - BIT14      ;DRIVE UNSAFE
568      020000      OPI      = BIT13      ;OPERATION INCOMPLETE
569      010000      DTE      - BIT12      ;DRIVE TIMING ERROR
570      004000      WLE      = BIT11      ;WRITE LOCK ERROR
571      002000      IAE      = BIT10      ;INVALID ADDRESS ERROR
572      001000      AOE      = BIT09      ;ADDRESS OVERFLOW ERROR
573      000400      HCRC     - BIT08      ;HEADER CRC ERROR
574      000200      HCE      - BIT07      ;HEADER COMPARE ERROR
575      000100      ECH      = BIT06      ;ECC "HARD" ERROR
576      000040      WCF      = BIT05      ;WRITE CLOCK FAILURE
577      000020      FER      - BIT04      ;FORMAT ERROR
578      000010      PAR      - BIT03      ;PARITY ERROR
579      000004      RMR      - BIT02      ;REGISTER MODIFICATION REFUSED
580      000002      ILR      = BIT01      ;ILLEGAL REGISTER
581      000001      ILF      = BIT00      ;ILLEGAL FUNCTION
582
583      115760      NDTMSK   - DCK.DTE!WLE.AOE.HCRC.HCE!ECH.WCF!FER
584      ;'NDTMSK' IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
585      ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS
586
587      ;*RMAS ATTENTION SUMMARY REGISTER
588
589      000377      ATNMSK   - 377      ;MASK FOR ATTENTION BITS
590
591      ;*RMLA LOOK AHEAD REGISTER
592
593      002000      SC4      - BIT10      ;SECTOR COUNT 16
594      001000      SC3      = BIT09      ;SECTOR COUNT 8
595      000400      SC2      = BIT08      ;SECTOR COUNT - 4
596      000200      SC1      - BIT07      ;SECTOR COUNT - 2
597      000100      SC0      = BIT06      ;SECTOR COUNT - 1
598
599      003700      SCTMSK   - 003700   ;SECTOR COUNT MASK
600
601      ;*RMMR1 MAINTENANCE REGISTER #1
602
603      ;WRITE ONLY BITS
604      100000      DBCK     = BIT15      ;DEBUG CLOCK
605      040000      DBEN     = BIT14      ;DEBUG CLOCK ENABLE
606      020000      DEBL     = BIT13      ;DIAGNOSTIC END OF BLOCK
607      010000      DTO      = BIT12      ;DIAGNOSTIC TIMEOUT
608      004000      MCLK     = BIT11      ;MAINTENANCE CLOCK
609      002000      MRD      = BIT10      ;READ DATA
610      001000      MUR      = BIT09      ;UNIT READY
611      000400      MOC      = BIT08      ;ON CYLINDER
612      000200      MSER     = BIT07      ;SEEK ERROR
613      000100      MDF      = BIT06      ;DRIVE FAULT
614      000040      MS       = BIT05      ;SECTOR PULSE
615      000010      MWP      - BIT03      ;WRITE PROTECT
616      000004      MI       = BIT02      ;INDEX PULSE
617      000002      MSC      = BIT01      ;SECTOR COMPARE
618      000001      DMD      - BIT00      ;DIAGNOSTIC MODE
619
620      ;READ ONLY BITS
621      100000      OCC      - BIT15      ;OCCUPIED

```

622	040000	RG	= BIT14	:RUN AND GO
623	020000	EBL	- BIT13	:END OF BLOCK
624	010000	REX	- BIT12	:EXCEPTION
625	004000	ESRC	- BIT11	:ENABLE SEARCH
626	002000	PLFS	= BIT10	:LOOKING FOR SYNC
627	001000	ECRC	- BIT09	:ENABLE CRC OUT
628	000400	PDA	- BIT08	:DATA AREA
629	000200	PHA	- BIT07	:HEADER AREA
630	000100	CONT	- BIT06	:CONTINUE
631	000040	WC	= BIT05	:WORD CLOCK
632	000020	EECC	- BIT04	:ENABLE ECC OUT
633	000010	MWD	- BIT03	:WRITE DATA BIT
634	000004	LS	- BIT02	:LAST SECTOR
635	000002	LST	- BIT01	:LAST SECTOR AND TRACK
636	000001	DMD	= BIT00	:DIAGNOSTIC MODE
637				
638		;*RMDT DRIVE TYPE REGISTER		
639				
640	100000	NSA	BIT15	:NOT SECTOR ADDRESSED 0
641	040000	TAP	- BIT14	:TAPE DRIVE = 0
642	020000	MOH	= BIT13	:MOVING HEAD = 1
643	004000	DRQ	- BIT11	:DRIVE REQUEST REQUIRED
644				
645	020024	SNGPRT	- 020024	:SINGLE PORT DRIVE TYPE (RM02)
646	024024	DULPRT	- 024024	:DUAL PORT DRIVE TYPE (RM02)
647				
648		;*RMOF OFFSET REGISTER		
649				
650	010000	FMT16	- BIT12	:16 BIT WORD FORMAT
651	004000	ECI	BIT11	:ECC INHIBIT
652	002000	HCI	- BIT10	:HEADER COMPARE INHIBIT
653	000200	OFD	- BIT07	:OFFSET FORWARD
654				
655		;*RMDC DESIRED CYLINDER ADDRESS REGISTER		
656				
657	001777	CYLMSK	- 001777	:MASK FOR CYLINDER ADDRESS
658				
659		;*RMMR2 MAINTENANCE REGISTER #2		
660				
661		:READ ONLY BITS		
662	100000	RQA	- BIT15	:PORT A REQUEST
663	040000	RQB	BIT14	:PORT B REQUEST
664	020000	TAG	= BIT13	:TAG CONTROL
665	010000	TST	- BIT12	:COMMAND SEQUENCE TEST BIT
666	004000	CC	- BIT11	:CONTROL OR CYLINDER TAG
667	002000	CH	- BIT10	:CONTROL OR HEAD TAG
670	001000	BB09	- BIT09	:TAG BUS
	000400	BB08	- BIT08	:TAG BUS
	000200	BB07	- BIT07	:TAG BUS
	000100	BB06	- BIT06	:TAG BUS
	000040	BB05	= BIT05	:TAG BUS
	000020	BB04	- BIT04	:TAG BUS
	000010	BB03	= BIT03	:TAG BUS
	000004	BB02	- BIT02	:TAG BUS
	000002	BB01	= BIT01	:TAG BUS
	000001	BB00	BIT00	:TAG BUS

```

672          ;*RMER2 ERROR REGISTER 2
673
674          100000      BSE      - BIT15      ;BAD SECTOR ERROR
675          040000      SKI      - BIT14      ;SEEK INCOMPLETE
676          020000      OPE      - BIT13      ;OPERATOR PLUG ERROR
677          010000      IVC      - BIT12      ;INVALID COMMAND ERROR
678          004000      LSC      - BIT11      ;LOSS OF SYSTEM CLOCK
679          002000      LBC      - BIT10      ;LOSS OF BIT CLOCK
680          000200      DVC      - BIT07      ;DEVICE CHECK
681          000010      DPE      = BIT03      ;DATA PARITY ERROR
682
683          .SBTTL  PROGRAM MNEMONICS
684
685          100000      MSE      - BIT15      ;MANUFACTURING DETECTED SECTOR ERROR
686          040000      USE      - BIT14      ;USER DETECTED SECTOR ERROR
687
688          .SBTTL  RM REGISTER INDEX VALUES
689
690          000000      RMCS1   - 00          ;CONTROL STATUS REGISTER #1
691          000006      RMDA    - 06          ;DISK ADDRESS REGISTER
692          000012      RMDS    - 12          ;DRIVE STATUS REGISTER
693          000014      RMER1   - 14          ;ERROR REGISTER #1
694          000016      RMAS    = 16          ;ATTENTION SUMMARY REGISTER
695          000020      RMLA    = 20          ;LOOK AHEAD REGISTER
696          000024      RMMR1   - 24          ;MAINTENANCE REGISTER
697          000026      RMDT    - 26          ;DRIVE TYPE REGISTER
698          000030      RMSN    = 30          ;SERIAL NUMBER REGISTER
699          000032      RMGF    - 32          ;OFFSET REGISTER
700          000034      RMDC    = 34          ;DESIRED CYLINDER REGISTER
701          000036      RMHR    = 36          ;HOLDING REGISTER
702          000040      RMMR2   = 40          ;MAINTENANCE REGISTER #2
703          000042      RMER2   - 42          ;ERROR REGISTER #2
704          000044      RMEC1   = 44          ;ECC POSITION REGISTER
705          000046      RMEC2   = 46          ;ECC PATTERN REGISTER
706          000050      ILRG50   = 50          ;ILLEGAL REGISTER 50
707          000052      ILRG52   = 52          ;ILLEGAL REGISTER 52
708          000054      ILRG54   = 54          ;ILLEGAL REGISTER 54
709          000056      ILRG56   - 56          ;ILLEGAL REGISTER 56
710          000060      ILRG60   = 60          ;ILLEGAL REGISTER 60
711          000062      ILRG62   = 62          ;ILLEGAL REGISTER 62
712          000064      ILRG64   - 64          ;ILLEGAL REGISTER 64
713          000066      ILRG66   - 66          ;ILLEGAL REGISTER 66
714          000070      ILRG70   = 70          ;ILLEGAL REGISTER 70
715          000072      ILRG72   = 72          ;ILLEGAL REGISTER 72
716          000074      ILRG74   = 74          ;ILLEGAL REGISTER 74
717          000076      ILRG76   = 76          ;ILLEGAL REGISTER 76
718
719          000077      IDXMSK   77          ;MASK FOR REGISTER INDEX NUMBER
720
721          .SBTTL  RM CONTROLLER REGISTER BIT DEFINITIONS
722
723          ;*RMCS1 CONTROL STATUS REGISTER #1
724
725          100000      SC      - BIT15      ;SPECIAL CONDITION-READ ONLY
726          040000      TRE      - BIT14      ;TRANSFER ERROR
727          020000      MCPE     - BIT13      ;MASSBUS CONTROL BUS PARITY ERROR-READ ONLY
728          002000      PSEL     - BIT10      ;PORT B SELECT

```

720	001000	A17	= BIT09	:ADDRESS EXTENSION
721	000400	A16	= BIT08	:ADDRESS EXTENSION
722	000200	RDY	= BIT07	:READY-READ ONLY
723	000100	IE	= BIT06	:INTERRUPT ENABLE
724				
725		;*RMCS2 RH CONTROL STATUS REGISTER #2		
726				
727	100000	DLT	= BIT15	:DATA LATE-READ ONLY
728	040000	WCE	= BIT14	:WRITE CHECK ERROR-READ ONLY
729	020000	UPE	= BIT13	:UNIBUS PARITY ERROR
730	010000	NED	= BIT12	:NONEXISTANT DRIVE-READ ONLY
731	004000	NEM	= BIT11	:NONEXISTANT MEMORY-READ ONLY
732	002000	PGE	= BIT10	:PROGRAM ERROR-READ ONLY
733	001000	MXF	= BIT09	:MISSED TRANSFER
734	000400	MDPE	= BIT08	:MASSBUS DATA BUS PARITY ERROR-READ ONLY
735	000200	OR	= BIT07	:OUTPUT READY-READ ONLY
736	000100	IR	= BIT06	:INPUT READY-READ ONLY
737	000040	CLR	= BIT05	:CONTROLLER CLEAR
738	000020	PAT	= BIT04	:PARITY TEST
739	000010	BAI	= BIT03	:UNIBUS ADDRESS INCREMENT INHIBIT
742	000004	U2	= BIT02	:UNIT SELECT
	000002	U1	= BIT01	:UNIT SELECT
	000001	U0	= BIT00	:UNIT SELECT
743				
744		:UNIT SELECT MASK		
745				
746	000007	UNTMSK	= 7	:UNIT SELECT MASK
747				
748		;*RMCS3 RH70 CONTROL STATUS REGISTER #3		
749				
750	100000	APE	= BIT15	:ADDRESS PARITY ERROR
751	040000	DPEHI	= BIT14	:DATA PARITY ERROR HIGH WORD
752	020000	DPELO	= BIT13	:DATA PARITY ERROR LOW WORD
753	010000	WCEHI	= BIT12	:WRITE CHECK ERROR HIGH WORD
754	004000	WCELO	= BIT11	:WRITE CHECK ERROR LOW WORD
755	002000	DBL	= BIT10	:DOUBLE WORD TRANSFER
756	000100	IE	= BIT06	:INTERRUPT ENABLE
757	000010	IPCK3	= BIT03	:INVERT PARITY CHECK
758	000004	IPCK2	= BIT02	:INVERT PARITY CHECK
759	000002	IPCK1	= BIT01	:INVERT PARITY CHECK
760	000001	IPCK0	= BIT00	:INVERT PARITY CHECK
761				
762		.SBTTL RH CONTROLLER REGISTER INDEX VALUES		
763				
764	000000	RMCS1	= 00	:CONTROL, STATUS REGISTER #1
765	000002	RMWC	= 02	:WORD COUNT REGISTER
766	000004	RMBA	= 04	:BUS ADDRESS REGISTER
767	000010	RMCS2	= 10	:CONTROL, STATUS REGISTER #2
768	000022	RMDB	= 22	:DATA BUFFER
769	000050	RMBAE	= 50	:BUS ADDRESS EXTENSION
770	000052	RMCS3	= 52	:CONTROL, STATUS REGISTER #3
771				
772	176700	ABASE	= 176700	:UNIBUS ADDRESS
773	120254	AVECT1	= 120254	:UNIBUS VECTOR ADDRESS AND PRIORITY
774				


```

1          .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          .=174
000174 000000  DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
000176 000000  SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
          :
          .SBTTL STARTING ADDRESS(ES)
2          000200 000137 005432          JMP @#START          ;;JUMP TO STARTING ADDRESS OF PROGRAM
3          000204 000137 005422          JMP @#START1         ;CHANGE RH/RM BUS ADDRESS
4
5          .SBTTL ACT11 HOOKS
          ;*****
          ;HOOKS REQUIRED BY ACT11
          000210          $SVPC=.          ;SAVE PC
          000046 000046          .=46
          000046 032122          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
          000052 000052          .=52
          000052 000000          .WORD 0          ;;2)SET LOC.52 TO ZERO
          000210 000210          .=$$VPC          ;; RESTORE PC
6
7          001100          .=1100
8          .SBTTL APT PARAMETER BLOCK
          ;*****
          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
          ;*****
          001100          .$X=.          ;;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 001222          $MBADR: .WORD $MAIL          ;;ADDRESS OF APT MAILBOX (BITS 0-15)
          001104 000006          $TSTM: .WORD 6          ;;RUN TIM OF LONGEST TEST
          001106 000006          $PASTM: .WORD 6          ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
          001110 000006          $UNITM: .WORD 6          ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
          001112 000042          .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
9          001114          TAGADR .
  
```

0

.SBTTL COMMON TAGS

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

001114	001114			.=TAGADR	
001114	000000			\$CMTAG: .WORD 0	::START OF COMMON TAGS
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			\$BCDAT: .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			SWR: .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			\$TMP0: .WORD 0	::USER DEFINED
001176	000000			\$TMP1: .WORD 0	::USER DEFINED
001200	000000			\$TMP2: .WORD 0	::USER DEFINED
001202	000000			\$TMP3: .WORD 0	::USER DEFINED
001204	000000			\$TMP4: .WORD 0	::USER DEFINED
001206	000000			\$TIMES: 0	::MAX. NUMBER OF ITERATIONS
001210	000000			\$ESCAPE: 0	::ESCAPE ON ERROR ADDRESS
001212	207	377	377	\$BELL: .ASCIZ <207><377><377>	::CODE FOR BELL
001216	077			\$QUES: .ASCII /?/	::QUESTION MARK
001217	015			\$CRLF: .ASCII <15>	::CARRIAGE RETURN
001220	012	000		\$LF: .ASCIZ <12>	::LINE FEED

.SBTTL APT MAILBOX-ETABLE

001222				.EVEN	
001222	000000			\$MAIL: .WORD	::APT MAILBOX
001224	000000			\$MSGTY: .WORD	AMSGTY ::MESSAGE TYPE CODE
001226	000000			\$FATAL: .WORD	AFATAL ::FATAL ERROR NUMBER
				\$TESTN: .WORD	ATESTN ::TEST NUMBER

001230	000000	\$PASS:	.WORD	APASS	::PASS COUNT
001232	000000	\$DEVCT:	.WORD	ADEVCT	::DEVICE COUNT
001234	000000	\$UNIT:	.WORD	AUNIT	::I/O UNIT NUMBER
001236	000000	\$MSGAD:	.WORD	AMSGAD	::MESSAGE ADDRESS
001240	000000	\$MSGLG:	.WORD	AMSGLG	::MESSAGE LENGTH
001242		\$ETABLE:			::APT ENVIRONMENT TABLE
001242	000	\$ENV:	.BYTE	AENV	::ENVIRONMENT BYTE
001243	000	\$ENVM:	.BYTE	AENVM	::ENVIRONMENT MODE BITS
001244	000000	\$SWREG:	.WORD	ASWREG	::APT SWITCH REGISTER
001246	000000	\$USWR:	.WORD	AUSWR	::USER SWITCHES
001250	000000	\$CPUOP:	.WORD	ACPUOP	::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
001252	000	\$MAMS1:	.BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
001253	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
001254	000000	\$MADR1:	.WORD	AMADR1	::HIGH ADDRESS,BLK#1
		*			MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001256	000	\$MAMS2:	.BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
001257	000	\$MTYP2:	.BYTE	AMTYP2	::MEM. TYPE,BLK#2
001260	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
001262	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
001263	000	\$MTYP3:	.BYTE	AMTYP3	::MEM. TYPE,BLK#3
001264	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
001266	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
001267	000	\$MTYP4:	.BYTE	AMTYP4	::MEM. TYPE,BLK#4
001270	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
001272	120254	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
001274	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
001276	176700	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
001300	000000	\$DEVW:	.WORD	ADEVW	::DEVICE MAP
001302	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
001304	000000	\$CDW2:	.WORD	ACDW2	::CONTROLLER DESCRIPTION WORD#2
001306	000000	\$DDW0:	.WORD	ADDW0	::DEVICE DESCRIPTOR WORD#0
001310	000000	\$DDW1:	.WORD	ADDW1	::DEVICE DESCRIPTOR WORD#1
001312	000000	\$DDW2:	.WORD	ADDW2	::DEVICE DESCRIPTOR WORD#2
001314	000000	\$DDW3:	.WORD	ADDW3	::DEVICE DESCRIPTOR WORD#3
001316	000000	\$DDW4:	.WORD	ADDW4	::DEVICE DESCRIPTOR WORD#4
001320	000000	\$DDW5:	.WORD	ADDW5	::DEVICE DESCRIPTOR WORD#5
001322	000000	\$DDW6:	.WORD	ADDW6	::DEVICE DESCRIPTOR WORD#6
001324	000000	\$DDW7:	.WORD	ADDW7	::DEVICE DESCRIPTOR WORD#7
001326		\$ETEND:			
		.MEXIT			

.SBTTL USER DEFINED TAGS

001326	000000	CTFLG:	.WORD	0	:CONTAINS CONTROL-C FLAG
001330	000000	CHGADR:	.WORD	0	:CHANGE RH/RM BUS ADDRESS = -1, NO CHANGE = 0
001332	000000	XXDP:	.WORD	0	:THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH :THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE : 'XXDP' DEVICE CODE FOR THE RM05/3/2.
001334	000	LSTRK:	.BYTE	0	:LO BYTE = 0
001335	000		.BYTE	0	:HI BYTE, CONTAINS LAST TRACK ADDRESS OF UNIT :UNDER TEST. RM02/3 = 4., RM05 = 18.

:THE REGISTER INPUT BUFFER IS USED FOR
:STORING DRIVE STATUS

001336

GETBUF:

001336	000000	:REGISTER INPUT BUFFER			
001340	000000	RMCS1I:	.WORD	0	:CONTROL, STATUS REGISTER #1
001342	000000	RMWCI:	.WORD	0	:WORD COUNT REGISTER
001344	000000	RMBAI:	.WORD	0	:BUS ADDRESS REGISTER
001346	000000	RMDAI:	.WORD	0	:DISK ADDRESS REGISTER
001350	000000	RMCS2I:	.WORD	0	:CONTROL, STATUS REGISTER #2
001352	000000	RMDSI:	.WORD	0	:DRIVE STATUS REGISTER
001354	000000	RMER1I:	.WORD	0	:ERROR REGISTER #1
001356	000000	RMASI:	.WORD	0	:ATTENTION SUMMARY REGISTER
001360	000000	RMLAI:	.WORD	0	:LOOK AHEAD REGISTER
001362	000000	RMDBI:	.WORD	0	:DATA BUFFER
001364	000000	RMMR1I:	.WORD	0	:MAINTENANCE REGISTER #1
001366	000000	RMDTI:	.WORD	0	:DRIVE TYPE REGISTER
001370	000000	RMSNI:	.WORD	0	:SERIAL NUMBER REGISTER
001372	000000	RMOFI:	.WORD	0	:OFFSET REGISTER
001374	000000	RMDCI:	.WORD	0	:DESIRED CYLINDER REGISTER
001376	000000	RMHRI:	.WORD	0	:HOLDING REGISTER
001400	000000	RMMR2I:	.WORD	0	:MAINTENANCE REGISTER #2
001402	000000	RMER2I:	.WORD	0	:ERROR REGISTER #2
001404	000000	RMEC1I:	.WORD	0	:ECC POSITION REGISTER
001406	000000	RMEC2I:	.WORD	0	:ECC PATTERN REGISTER
001410	000000	RMBAEI:	.WORD	0	:BUS ADDRESS EXTENSION REGISTER
		RMCS3I:	.WORD	0	:CONTROL, STATUS REGISTER #3

:THE REGISTER OUTPUT BUFFER IS USED FOR
:ASSEMBLING DATA GOING TO REGISTER

001412

PUTBUF:

001412	000000	:REGISTER OUTPUT BUFFER			
001414	000000	RMCS1O:	.WORD	0	:CONTROL, STATUS REGISTER #1
001416	000000	RMWCO:	.WORD	0	:WORD COUNT REGISTER
001420	000000	RMBAO:	.WORD	0	:BUS ADDRESS REGISTER
001422	000000	RMDAO:	.WORD	0	:DISK ADDRESS REGISTER
001424	000000	RMCS2O:	.WORD	0	:CONTROL, STATUS REGISTER #2
001426	000000	RMDSO:	.WORD	0	:DRIVE STATUS REGISTER
001430	000000	RMER1O:	.WORD	0	:ERROR REGISTER #1
001432	000000	RMASO:	.WORD	0	:ATTENTION SUMMARY REGISTER
001434	000000	RMLAO:	.WORD	0	:LOOK AHEAD REGISTER
001436	000000	RMDBO:	.WORD	0	:DATA BUFFER
		RMMR1O:	.WORD	0	:MAINTENANCE REGISTER #1

001440 000000
 001442 000000
 001444 000000
 001446 000000
 001450 000000
 001452 000000
 001454 000000
 001456 000000
 001460 000000
 001462 000000
 001464 000000

RMDTO: .WORD 0 ;DRIVE TYPE REGISTER
 RMSNO: .WORD 0 ;SERIAL NUMBER REGISTER
 RMGFO: .WORD 0 ;OFFSET REGISTER
 RMDCO: .WORD 0 ;DESIRED CYLINDER REGISTER
 RMHRO: .WORD 0 ;HOLDING REGISTER
 RMMR20: .WORD 0 ;MAINTENANCE REGISTER #2
 RMER20: .WORD 0 ;ERROR REGISTER #2
 RMEC10: .WORD 0 ;ECC POSITION REGISTER
 RMEC20: .WORD 0 ;ECC PATTERN REGISTER
 RMBAE0: .WORD 0 ;BUS ADDRESS EXTENSION REGISTER
 RMCS30: .WORD 0 ;CONTROL, STATUS REGISTER #3

;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
 ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. THE
 ;FIRST WORD CONTAINS THE ADDRESS OF THE DEVICE UNDER TEST
 ;IN THE TABLE. A ZERO WORD IS A BLANK AND REPRESENTS THE
 ;END OF THE QUE.

001466 000000
 001470 000000
 001510 000000

TSTQJE: .WORD 0 ;CONTAINS DEVICE POINTER
 .BLKW 8. ;TEST QUE FOR DEVICES UNDER TEST
 .WORD 0 ;TABLE TERMINATOR GOES HERE WHEN
 ;ALL 8. DEVICES ARE UNDER TEST.

001512 000000

;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
 ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
 MEDENB: .WORD 0 ;MEDIA ENABLE

001514 000000
 001516 000000
 001520 000000
 001522 000000

;LOCATIONS 'ASNDC' AND 'ASNDC' CONTAIN THE CYLINDER, TRACK AND SECTOR
 ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
 ASNDC: .WORD 0 ;ASSIGNED DESIRED CYLINDER
 ASNDA: .WORD 0 ;ASSIGNED TRACK, AND SECTOR
 CLKADR: .WORD 0 ;UNIBUS ADDRESS OF KW11 CLOCK
 CLKVCT: .WORD 0 ;VECTOR ADDRESS OF KW11 CLOCK

001524

;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
 ;ARE READ BY THE GET SUBROUTINE. THE LIST IS TERMINATED BY
 ;A NEGATIVE BYTE.
 GETINX: .BLKB 23. ;GET INDEX TABLE

001553

;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
 ;ARE WRITTEN BY THE PUT SUBROUTINE. THE LIST IS TERMINATED BY
 ;A NEGATIVE BYTE.
 PUTINX: .BLKB 23. ;PUT INDEX TABLE

;PUT TAGS HERE

.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 (\$ERRPT).
;*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

Index	Item 1	Item 2	Pointer 1	Pointer 2	Pointer 3	Pointer 4	Description
1	001602						
2							
3	001602	065324	EMT1	EHT1	EDT1	EFT1	WRONG UNIT SELECTED
	001604	071410					
	001606	071534					
	001610	071624					
4							
5							
6	001612	065330	EMT2	EHT1	EDT1	EFT1	DEVICE WENT UNAVAILABLE
	001614	071410					
	001616	071534					
	001620	071624					
7							
8							
9	001622	065336	EMT3	EHT1	EDT1	EFT1	DEVICE WENT NONEXISTENT
	001624	071410					
	001626	071534					
	001630	071624					
10							
11							
12	001632	065344	EMT4	EHT1	EDT1	EFT1	CONTROLLER NOT READY
	001634	071410					
	001636	071534					
	001640	071624					
13							
14							
15	001642	065352	EMT5	EHT1	EDT1	EFT1	DRIVE NOT READY AND GO NOT RESET
	001644	071410					
	001646	071534					
	001650	071624					
16							
17							
							UNEXPECTED VALUE FOR "ATA" STATUS

Line	Address	Value	Label	Description
18	001652	065360	EMT6	
	001654	071410	EHT1	
	001656	071534	EDT1	
	001660	071624	EFT1	
19				
20			:ERROR 7	BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
21	001662	065366	EMT7	
	001664	000000	0	
	001666	000000	0	
	001670	000000	0	
22				
23			:ERROR 10	DRIVE NOT READY BUT GO IS RESET
24	001672	065374	EMT10	
	001674	071410	EHT1	
	001676	071534	EDT1	
	001700	071624	EFT1	
25				
26			:ERROR 11	GO NOT RESE* BUT DRIVE IS READY
27	001702	065400	EMT11	
	001704	071410	EHT1	
	001706	071534	EDT1	
	001710	071624	EFT1	
28				
29			:ERROR 12	INCORRECT FUNCTION CODE
30	001712	065404	EMT12	
	001714	071410	EHT1	
	001716	071534	EDT1	
	001720	071624	EFT1	
31				
32			:ERROR 13	PARITY ERROR READING REMOTE REGISTERS
33	001722	065412	EMT13	
	001724	071410	EHT1	
	001726	071534	EDT1	
	001730	071624	EFT1	
34				
35			:ERROR 14	TRANSFER ERROR IS INCORRECT
36	001732	065424	EMT14	
	001734	071410	EHT1	
	001736	071534	EDT1	
	001740	071624	EFT1	
37				
38			:ERROR 15	INCORRECT WORD COUNT
39				

ERROR POINTER TABLE

	001742	065432		EMT15
	001744	071410		EHT1
	001746	071534		EDT1
	001750	071624		EFT1
40				
41			:ERROR 16	INCORRECT BUS ADDRESS
42				
	001752	065440		EMT16
	001754	071410		EHT1
	001756	071534		EDT1
	001760	071624		EFT1
43				
44			:ERROR 17	INCORRECT LBT STATUS
45				
	001762	065450		EMT17
	001764	071410		EHT1
	001766	071534		EDT1
	001770	071624		EFT1
46				
47			:ERROR 20	INCORRECT AOE
48				
	001772	065460		EMT20
	001774	071410		EHT1
	001776	071534		EDT1
	002000	071624		EFT1
49				
50			:ERROR 21	INCORRECT DISK ADDRESS
51				
	002002	065470		EMT21
	002004	071410		EHT1
	002006	071534		EDT1
	002010	071624		EFT1
52				
53			:ERROR 22	INCORRECT CYLINDER ADDRESS
54				
	002012	065500		EMT22
	002014	071410		EHT1
	002016	071534		EDT1
	002020	071624		EFT1
55				
56			:ERROR 23	INCORRECT WLE STATUS
57				
	002022	065510		EMT23
	002024	071410		EHT1
	002026	071534		EDT1
	002030	071624		EFT1
58				
59			:ERROR 24	INCORRECT UPE STATUS
60				
	002032	065520		EMT24

	002034	071410	EHT1	
	002036	071534	EDT1	
	002040	071624	EFT1	
61				
62				:ERROR 25 INCORRECT WCF STATUS
63				
	002042	065530	EMT25	
	002044	071410	EHT1	
	002046	071534	EDT1	
	002050	071624	EFT1	
64				
65				:ERROR 26 INCORRECT WCE STATUS
66				
	002052	065540	EMT26	
	002054	071410	EHT1	
	002056	071534	EDT1	
	002060	071624	EFT1	
67				
68				:ERROR 27 INCORRECT MDPE STATUS
69				
	002062	065550	EMT27	
	002064	071410	EHT1	
	002066	071534	EDT1	
	002070	071624	EFT1	
70				
71				:ERROR 30 INCORRECT DCK STATUS
72				
	002072	065560	EMT30	
	002074	071410	EHT1	
	002076	071534	EDT1	
	002100	071624	EFT1	
73				
74				:ERROR 31 INCORRECT ECH STATUS
75				
	002102	065570	EMT31	
	002104	071410	EHT1	
	002106	071534	EDT1	
	002110	071624	EFT1	
76				
77				:ERROR 32 DLT SHOULD NOT BE SET
78				
	002112	065600	EMT32	
	002114	071410	EHT1	
	002116	071534	EDT1	
	002120	071624	EFT1	
79				
80				:ERROR 33 MXF SHOULD NOT BE SET
81				
	002122	065610	EMT33	
	002124	071410	EHT1	

ERROR POINTER TABLE

	002126	071534		EDT1
	002130	071624		EFT1
82				
83			;ERROR 34	DTE SHOULD NOT BE SET
84				
	002132	065620		EMT34
	002134	071410		EHT1
	002136	071534		EDT1
	002140	071624		EFT1
85				
86			;ERROR 35	INCORRECT HCRC STATUS
87				
	002142	065630		EMT35
	002144	071410		FHT1
	002146	071534		EDT1
	002150	071624		EFT1
88				
89			;ERROR 36	INCORRECT HCE STATUS
90				
	002152	065640		EMT36
	002154	071410		EHT1
	002156	071534		EDT1
	002160	071624		EFT1
91				
92			;ERROR 37	INCORRECT FER STATUS
93				
	002162	065650		EMT37
	002164	071410		EHT1
	002166	071534		EDT1
	002170	071624		EFT1
94				
95			;ERROR 40	DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
96				
	002172	065660		EMT40
	002174	071410		EHT1
	002176	071534		EDT1
	002200	071624		EFT1
97				
98			;ERROR 41	LOST 'MOL' DURING PACK ACKNOWLEDGE
99				
	002202	065666		EMT41
	002204	071410		EHT1
	002206	071534		EDT1
	002210	071624		EFT1
100				
101			;ERROR 42	UNSAFE ERROR DURING PACK ACKNOWLEDGE
102				
	002212	065676		EMT42
	002214	071410		EHT1
	002216	071534		EDT1

	002220	071624	EFT1	
103				
104			:ERROR 43	'DPI' ERROR DURING PACK ACKNOWLEDGE
105				
	002222	065710	EMT43	
	002224	071410	EHT1	
	002226	071534	EDT1	
	002230	071624	EFT1	
106				
107			:ERROR 44	'RMR' ERROR DURING PACK ACKNOWLEDGE
108				
	002232	065720	EMT44	
	002234	071410	EHT1	
	002236	071534	EDT1	
	002240	071624	EFT1	
109				
110			:ERROR 45	'ILR' ERROR DURING PACK ACKNOWLEDGE
111				
	002242	065730	EMT45	
	002244	071410	EHT1	
	002246	071534	EDT1	
	002250	071624	EFT1	
112				
113			:ERROR 46	'ILF' ERROR DURING PACK ACKNOWLEDGE
114				
	002252	065740	EMT46	
	002254	071410	EHT1	
	002256	071534	EDT1	
	002260	071624	EFT1	
115				
116			:ERROR 47	COMPOSITE ERROR STATUS IS INCORRECT
117				
	002262	065750	EMT47	
	002264	071410	EHT1	
	002266	071534	EDT1	
	002270	071624	EFT1	
118				
119			:ERROR 50	PARITY ERROR WRITING REMOTE REGISTERS
120				
	002272	065756	EMT50	
	002274	071410	EHT1	
	002276	071534	EDT1	
	002300	071624	EFT1	
121				
122			:ERROR 51	INCORRECT IAE STATUS DURING SEEK COMMAND
123				
	002302	065766	EMT51	
	002304	071410	EHT1	
	002306	071534	EDT1	
	002310	071624	EFT1	

124				
125				
126				
	002312	066000		
	002314	071410		
	002316	071534		
	002320	071624		
127				
128				
129				
130				
	002322	066016		
	002324	071410		
	002326	071534		
	002330	071624		
131				
132				
133				
	002332	066034		
	002334	071410		
	002336	071534		
	002340	071624		
134				
135				
136				
	002342	066044		
	002344	071410		
	002346	071534		
	002350	071624		
137				
138				
139				
	002352	066056		
	002354	071410		
	002356	071534		
	002360	071624		
140				
141				
142				
	002362	066074		
	002364	071410		
	002366	071534		
	002370	071624		
143				
144				
145				
146				
	002372	066104		
	002374	071410		
	002376	071534		

;ERROR 52 OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE

EMT52
EHT1
EDT1
EFT1

;ERROR 53 OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
ON CYLINDER LATCH DIDN'T RESET

EMT53
EHT1
EDT1
EFT1

;ERROR 54 SEEK INCOMPLETE ERROR DURING SEEK COMMAND

EMT54
EHT1
EDT1
EFT1

;ERROR 55 DEVICE CHECK DURING SEEK COMMAND

EMT55
EHT1
EDT1
EFT1

;ERROR 56 PIP IS STILL SET AFTER SEEK - SKI IS RESET

EMT56
EHT1
EDT1
EFT1

;ERROR 57 ATA DID NOT SET DURING SEEK COMMAND

EMT57
EHT1
EDT1
EFT1

;ERROR 60 I. ERROR DURING SEEK COMMAND - LOST
VOLUME VALID

EMT60
EHT1
EDT1

	002400	071624	EFT1	
147				
148			:ERROR 61	ERRONEOUS IVC ERROR DURING SEEK COMMAND -
149			:	VOLUME VALID IS STILL SET
150				
	002402	066122	EMT61	
	002404	071410	EHT1	
	002406	071534	EDT1	
	002410	071624	EFT1	
151				
152			:ERROR 62	MOL IS ZERO, BUT OPI WAS NOT
153			:	REPORTED DURING SEEK COMMAND
154				
	002412	066142	EMT62	
	002414	071410	EHT1	
	002416	071534	EDT1	
	002420	071624	EFT1	
155				
156			:ERROR 63	UNUSED
157				
	002422	000000	0	
	002424	000000	0	
	002426	000000	0	
	002430	000000	0	
158				
159			:ERROR 64	DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
160				
	002432	066160	EMT64	
	002434	071410	EHT1	
	002436	071534	EDT1	
	002440	071624	EFT1	
161				
162			:ERROR 65	DRIVE EXECUTED A SEEK WITH ERROR SET
163				
	002442	066200	EMT65	
	002444	071410	EHT1	
	002446	071534	EDT1	
	002450	071624	EFT1	
164				
165			:ERROR 66	UNEXPECTED ERROR SET IN RMER1
166				
	002452	066220	EMT66	
	002454	071410	EHT1	
	002456	071534	EDT1	
	002460	071624	EFT1	
167				
168			:ERROR 67	UNEXPECTED ERROR SET IN RMER2
169				
	002462	066232	EMT67	
	002464	071410	EHT1	

	002466	071534	EDT1	
	002470	071624	EFT1	
170				
171				:ERROR 70 ERRONEOUS "IAE" ERROR DURING RECALIBRATE
172				
	002472	066244	EMT70	
	002474	071410	EHT1	
	002476	071534	EDT1	
	002500	071624	EFT1	
173				
174				:ERROR 71 "ILF" ERROR DURING RECALIBRATE
175				
	002502	066254	EMT71	
	002504	071410	EHT1	
	002506	071534	EDT1	
	002510	071624	EFT1	
176				
177				:ERROR 72 "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
178				
	002512	066264	EMT72	
	002514	071410	EHT1	
	002516	071534	EDT1	
	002520	071624	EFT1	
179				
180				:ERROR 73 "OPI" ERROR DURING RECALIBRATE BECAUSE ON
181				: CYLINDER DIDNT DROP
182				
	002522	066302	EMT73	
	002524	071410	EHT1	
	002526	071534	EDT1	
	002530	071624	EFT1	
183				
184				:ERROR 74 "IVC" ERROR DURING RECALIBRATE - "VV" = 0
185				
	002532	066320	EMT74	
	002534	071410	EHT1	
	002536	071534	EDT1	
	002540	071624	EFT1	
186				
187				:ERROR 75 ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" 1
188				
	002542	066330	EMT75	
	002544	071410	EHT1	
	002546	071534	EDT1	
	002550	071624	EFT1	
189				
190				:ERROR 76 "SKI" ERROR DURING RECALIBRATE
191				
	002552	066350	EMT76	
	002554	071410	EHT1	

ERROR POINTER TABLE

	002556	071534	EDT1	
	002560	071624	EFT1	
192				
193				;ERROR 77 'DVC' OCCURRED DURING RECALIBRATE
194				
	002562	066360	EMT77	
	002564	071410	EHT1	
	002566	071534	EDT1	
	002570	071624	EFT1	
195				
196				;ERROR 100 LOST 'MOL' DURING RECALIBRATE - 'DPI' = 0
197				
	002572	066372	EMT100	
	002574	071410	EHT1	
	002576	071534	EDT1	
	002600	071624	EFT1	
198				
199				;ERROR 101 LOST 'VV' DURING RECALIBRATE - 'IVC' = 0
200				
	002602	066410	EMT101	
	002604	071410	EHT1	
	002606	071534	EDT1	
	002610	071624	EFT1	
201				
202				;ERROR 102 'ATA' DID NOT SET DURING RECALIBRATE
203				
	002612	066426	EMT102	
	002614	071410	EHT1	
	002616	071534	EDT1	
	002620	071624	EFT1	
204				
205				;ERROR 103 'DM' DID NOT RESET DURING RECALIBRATE
206				
	002622	066436	EMT103	
	002624	071410	EHT1	
	002626	071534	EDT1	
	002630	071624	EFT1	
207				
208				;ERROR 104 'PIP' IS STIL SET AFTER RECALIBRATE
209				
	002632	066450	EMT104	
	002634	071410	EHT1	
	002636	071534	EDT1	
	002640	071624	EFT1	
210				
211				;ERROR 105 UNEXPECTED 'ILR' ERROR DURING RECALIBRATE
212				
	002642	066466	EMT105	
	002644	071410	EHT1	
	002646	071534	EDT1	

Line	Code	Pointer	Module	Description
	002650	071624	EFT1	
213				
214				
215				:ERROR 106 UNEXPECTED 'RMR' ERROR DURING RECALIBRATE
	002652	066476	EMT106	
	002654	071410	EHT1	
	002656	071534	EDT1	
	002660	071624	EFT1	
216				
217				
218				:ERROR 107 'UNS' ERROR DURING RECALIBRATE - AC POWER IS LOW
	002662	066506	EMT107	
	002664	071410	EHT1	
	002666	071534	EDT1	
	002670	071624	EFT1	
219				
220				
221				:ERROR 110 CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
	002672	066526	EMT110	
	002674	071432	EHT110	
	002676	071552	EDT110	
	002700	071642	EFT110	
222				
223				
224				:ERROR 111 NONEXISTENT DEVICE
	002702	066540	EMT111	
	002704	071436	EHT111	
	002706	071554	EDT111	
	002710	071644	EFT111	
225				
226				
227				:ERROR 112 DEVICE NOT AVAILABLE
	002712	066546	EMT112	
	002714	071436	EHT111	
	002716	071554	EDT111	
	002720	071644	EFT111	
228				
229				
230				:ERROR 113 BUS TIMEOUT-NED STATUS FAILURE
	002722	066554	EMT113	
	002724	000000	0	
	002726	000000	0	
	002730	000000	0	
231				
232				
233				:ERROR 114 DEVICE NOT AN RM05/3/2
	002732	066570	EMT114	
	002734	071442	EHT114	
	002736	071556	EDT114	
	002740	071646	EFT114	

234			
235		;ERROR	115 RMCS1 NOT INITIALIZED BY UNIBUS
236	002742 066576		EMT115
	002744 071410		EHT1
	002746 071534		EDT1
	002750 071624		EFT1
237			
238		;ERROR	116 RMBA NOT INITIALIZED BY UNIBUS
239	002752 066606		EMT116
	002754 071410		EHT1
	002756 071534		EDT1
	002760 071624		EFT1
240			
241		;ERROR	117 RMCS2 NOT INITIALIZED BY UNIBUS
242	002762 066616		EMT117
	002764 071410		EHT1
	002766 071534		EDT1
	002770 071624		EFT1
243			
244		;ERROR	120 RMER1 NOT INITIALIZED BY UNIBUS
245	002772 066626		EMT120
	002774 071410		EHT1
	002776 071534		EDT1
	003000 071624		EFT1
246			
247		;ERROR	121 RMAS NOT INITIALIZED BY UNIBUS
248	003002 066636		EMT121
	003004 071410		EHT1
	003006 071534		EDT1
	003010 071624		EFT1
249			
250		;ERROR	122 RMMR1 NOT INITIALIZED BY UNIBUS
251	003012 066646		EMT122
	003014 071410		EHT1
	003016 071534		EDT1
	003020 071624		EFT1
252			
253		;ERROR	123 RMDS NOT INITIALIZED BY UNIBUS
254	003022 066656		EMT123
	003024 071410		EHT1
	003026 071534		EDT1
	003030 071624		EFT1

Line	Code	Address	Message
255			
256	:ERROR	124	RMEC2 NOT INITIALIZED BY UNIBUS
257			
	003032	066666	EMT124
	003034	071410	EHT1
	003036	071534	EDT1
	003040	071624	EFT1
258			
259	:ERROR	125	RMTR2 NOT INITIALIZED BY UNIBUS
260			
	003042	066676	EMT125
	003044	071410	EHT1
	003046	071534	EDT1
	003050	071624	EFT1
261			
262	:ERROR	126	RMCS1 NOT CLEARED BY CONTROLLER CLEAR
263			
	003052	066706	EMT126
	003054	071410	EHT1
	003056	071534	EDT1
	003060	071624	EFT1
264			
265	:ERROR	127	RMBA NOT CLEARED BY CONTROLLER CLEAR
266			
	003062	066720	EMT127
	003064	071410	EHT1
	003066	071534	EDT1
	003070	071624	EFT1
267			
268	:ERROR	130	RMCS2 NOT CLEARED BY CONTROLLER CLEAR
269			
	003072	066732	EMT130
	003074	071410	EHT1
	003076	071534	EDT1
	003100	071624	EFT1
270			
271	:ERROR	131	RMER1 NOT CLEARED BY CONTROLLER CLEAR
272			
	003102	066744	EMT131
	003104	071410	EHT1
	003106	071534	EDT1
	003110	071624	EFT1
273			
274	:ERROR	132	RMAS NOT CLEARED BY CONTROLLER CLEAR
275			
	003112	066756	EMT132
	003114	071410	EHT1
	003116	071534	EDT1
	003120	071624	EFT1
276			

277			:ERROR 133	RMMR1 NOT CLEARED BY CONTROLLER CLEAR
278	003122	066770		
	003124	071410	EMT133	
	003126	071534	EHT1	
	003130	071624	EDT1	
			EFT1	
279				
280			:ERROR 134	RMDS NOT CLEARED BY CONTROLLER CLEAR
281	003132	067002		
	003134	071410	EMT134	
	003136	071534	EHT1	
	003140	071624	EDT1	
			EFT1	
282				
283			:ERROR 135	RMEC2 NOT CLEARED BY CONTROLLER CLEAR
284	003142	067014		
	003144	071410	EMT135	
	003146	071534	EHT1	
	003150	071624	EDT1	
			EFT1	
285				
286			:ERROR 136	RMMR2 NOT CLEARED BY CONTROLLER CLEAR
287	003152	067026		
	003154	071410	EMT136	
	003156	071534	EHT1	
	003160	071624	EDT1	
			EFT1	
288				
289			:ERROR 137	RMCS1 NOT CLEARED BY ERROR CLEAR
290	003162	067040		
	003164	071410	EMT137	
	003166	071534	EHT1	
	003170	071624	EDT1	
			EFT1	
291				
292			:ERROR 140	RMCS2 NOT CLEARED BY ERROR CLEAR
293	003172	067050		
	003174	071410	EMT140	
	003176	071534	EHT1	
	003200	071624	EDT1	
			EFT1	
294				
295			:ERROR 141	RMCS1 NOT CLEARED BY DRIVE CLEAR
296	003202	067060		
	003204	071410	EMT141	
	003206	071534	EHT1	
	003210	071624	EDT1	
			EFT1	
297				
298			:ERROR 142	RMDS NOT CLEARED BY DRIVE CLEAR

299

003212	067070	EMT142
003214	071410	EHT1
003216	071534	EDT1
003220	071624	EFT1

300

301

:ERROR 143 RMER1 NOT CLEARED BY DRIVE CLEAR

302

003222	067100	EMT143
003224	071410	EHT1
003226	071534	EDT1
003230	071624	EFT1

303

304

:ERROR 144 RMAS NOT CLEARED BY DRIVE CLEAR

305

003232	067110	EMT144
003234	071410	EHT1
003236	071534	EDT1
003240	071624	EFT1

306

307

:ERROR 145 RMMR1 NOT CLEARED BY DRIVE CLEAR

308

003242	067120	EMT145
003244	071410	EHT1
003246	071534	EDT1
003250	071624	EFT1

309

310

:ERROR 146 RMMR2 NOT CLEARED BY DRIVE CLEAR

311

003252	067130	EMT146
003254	071410	EHT1
003256	071534	EDT1
003260	071624	EFT1

312

313

:ERROR 147 RMER2 NOT CLEARED BY DRIVE CLEAR

314

003262	067140	EMT147
003264	071410	EHT1
003266	071534	EDT1
003270	071624	EFT1

315

316

:ERROR 150 RMEC2 NOT CLEARED BY DRIVE CLEAR

317

003272	067150	EMT150
003274	071410	EHT1
003276	071534	EDT1
003300	071624	EFT1

318

319

:ERROR 151 MEDIUM NOT ON LINE

320

ERROR POINTER TABLE				
003302	067160		EMT151	
003304	071410		EHT1	
003306	071534		EDT1	
003310	071624		EFT1	
321				
322		:ERROR 152		DRIVE FAULT
323				
003312	067172		EMT152	
003314	071410		EHT1	
003316	071534		EDT1	
003320	071624		EFT1	
324				
325		:ERROR 153		UNSAFE SHOULD BE SET BECAUSE DVC IS SET
326				
003322	067204		EMT153	
003324	071410		EHT1	
003326	071534		EDT1	
003330	071624		EFT1	
327				
328		:ERROR 154		UNSAFE SHOULD NOT BE SET, AC IS LOW
329				
003332	067222		EMT154	
003334	071410		EHT1	
003336	071534		EDT1	
003340	071624		EFT1	
330				
331		:ERROR 155		VOLUME VALID NOT SET BY PACK ACK
332				
003342	067240		EMT155	
003344	071410		EHT1	
003346	071534		EDT1	
003350	071624		EFT1	
333				
334		:ERROR 156		OFFSET MODE NOT SET BY OFFSET COMMAND
335				
003352	067252		EMT156	
003354	071410		EHT1	
003356	071534		EDT1	
003360	071624		EFT1	
336				
337		:ERROR 157		OFFSET MODE NOT RESET BY RTC COMMAND
338				
003362	067264		EMT157	
003364	071410		EHT1	
003366	071534		EDT1	
003370	071624		EFT1	
339				
340		:ERROR 160		RMOF NOT RESET BY RIP COMMAND
341				
003372	067276		EMT160	

ERROR POINTER TABLE				
	003374	071410		EMT1
	003376	071534		EDT1
	003400	071624		EFT1
342				
343			:ERROR 161	RMDA NOT RESET BY RIP COMMAND
344				
	003402	067306		EMT161
	003404	071410		EMT1
	003406	071534		EDT1
	003410	071624		EFT1
345				
346			:ERROR 162	RMDC NOT RESET BY RIP COMMAND
347				
	003412	067320		EMT162
	003414	071410		EMT1
	003416	071534		EDT1
	003420	071624		EFT1
348				
349			:ERROR 163	DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
350			:	WRITE BUFFER
351				
	003422	071204		EMT336
	003424	071472		EMT336
	003426	071570		EDT336
	003430	071660		EFT336
352				
353			:ERROR 164	OPI SHOULD NOT BE SET
354				
	003432	067342		EMT164
	003434	071410		EMT1
	003436	071534		EDT1
	003440	071624		EFT1
355				
356			:ERROR 165	IVC SHOULD NOT BE SET
357				
	003442	067350		EMT165
	003444	071410		EMT1
	003446	071534		EDT1
	003450	071624		EFT1
358				
359			:ERROR 166	IAE SHOULD NOT BE SET
360				
	003452	067356		EMT166
	003454	071410		EMT1
	003456	071534		EDT1
	003460	071624		EFT1
361				
362			:ERROR 167	NEM SHOULD NOT BE SET
363				
	003462	067364		EMT167

	003464	071410	EHT1	
	003466	071534	EDT1	
	003470	071624	EFT1	
364				
365				;ERROR 170 INCORRECT 'MOL' STATUS DURING DIAGNOSTIC MODE
366				
	003472	067372	EMT170	
	003474	071410	EHT1	
	003476	071534	EDT1	
	003500	071624	EFT1	
367				
368				;ERROR 171 'ATA' NOT SET DURING RETURN TO CENTERLINE
369				
	003502	067404	EMT171	
	003504	071410	EHT1	
	003506	071534	EDT1	
	003510	071624	EFT1	
370				
371				;ERROR 172 'ATA' NOT SET BY OFFSET COMMAND
372				
	003512	067414	EMT172	
	003514	071410	EHT1	
	003516	071534	EDT1	
	003520	071624	EFT1	
373				
374				;ERROR 173 RMER2 NOT INITIALIZED BY UNIBUS INIT
375				
	003522	067424	EMT173	
	003524	071410	EHT1	
	003526	071534	EDT1	
	003530	071624	EFT1	
376				
377				;ERROR 174 RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
378				
	003532	067434	EMT174	
	003534	071410	EHT1	
	003536	071534	EDT1	
	003540	071624	EFT1	
379				
380				;ERROR 175 SELECTED DEVICE IS IN WRITE PROTECT
381				
	003542	067446	EMT175	
	003544	071410	EHT1	
	003546	071534	EDT1	
	003550	071624	EFT1	
382				
383				;ERROR 176 CANNOT SET DIAGNOSTIC MODE
384				
	003552	067454	EMT176	
	003554	071410	EHT1	

	003556	071534	EDT1	
	003560	071624	EFT1	
385				
386			:ERROR	177 --RESERVED FOR POWER MONITOR BIT FAILURE--
387				
	003562	000000	0	
	003564	000000	0	
	003566	000000	0	
	003570	000000	0	
388				
389			:ERROR	200 INCORRECT 'PIP' STATUS DURING DIAGNOSTIC MODE
390				
	003572	067464	EMT200	
	003574	071410	EHT1	
	003576	071534	EDT1	
	003600	071624	EFT1	
391				
392			:ERROR	201 INCORRECT 'WRL' STATUS DURING DIAGNOSTIC MODE
393				
	003602	067476	EMT201	
	003604	071410	EHT1	
	003606	071534	EDT1	
	003610	071624	EFT1	
394				
395			:ERROR	202 INCORRECT 'SKI' STATUS DURING DIAGNOSTIC MODE
396				
	003612	067510	EMT202	
	003614	071410	EHT1	
	003616	071534	EDT1	
	003620	071624	EFT1	
397				
398			:ERROR	203 INCORRECT 'DVC' STATUS DURING DIAGNOSTIC MODE
399				
	003622	067522	EMT203	
	003624	071410	EHT1	
	003626	071534	EDT1	
	003630	071624	EFT1	
400				
401			:ERROR	204 'VV' WAS NOT RESET BY MAINTENANCE UNIT READY
402				
	003632	067534	EMT204	
	003634	071410	EHT1	
	003636	071534	EDT1	
	003640	071624	EFT1	
403				
404			:ERROR	205 SELECTED DEVICE HAS A PERSISTENT 'SKI' ERROR
405				
	003642	067552	EMT205	
	003644	071410	EHT1	
	003646	071534	EDT1	

	003650	071624	EFT1	
406				
407			:ERROR	206 'LBC' DID NOT SET DURING DIAGNOSTIC MODE
408				
	003652	067562	EMT206	
	003654	071410	EHT1	
	003656	071534	EDT1	
	003660	071624	EFT1	
409				
410			:ERROR	207 UNEXPECTED LOSS OF 'MOL' - MEDIUM IS OFF LINE
411				
	003662	067572	EMT207	
	003664	071410	EHT1	
	003666	071534	EDT1	
	003670	071624	EFT1	
412				
413			:ERROR	210 UNEXPECTED LOSS OF VOLUME VALID - 'VV' - 0
414				
	003672	067604	EMT210	
	003674	071410	EHT1	
	003676	071534	EDT1	
	003700	071624	EFT1	
415				
416			:ERROR	211 UNEXPECTED MECHANICAL MOTION - 'PIP' 1
417				
	003702	067612	EMT211	
	003704	071410	EHT1	
	003706	071534	EDT1	
	003710	071624	EFT1	
418				
419			:ERROR	212 UNEXPECTED DEVICE FAULT - 'DVC' = 1
420				
	003712	067626	EMT212	
	003714	071410	EHT1	
	003716	071534	EDT1	
	003720	071624	EFT1	
421				
422			:ERROR	213 UNEXPECTED SEEK INCOMPLETE ERROR - 'SKI' 1
423				
	003722	067642	EMT213	
	003724	071410	EHT1	
	003726	071534	EDT1	
	003730	071624	EFT1	
424				
425			:ERROR	214 DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
426				
	003732	067652	EMT214	
	003734	071422	EHT2	
	003736	071544	EDT2	
	003740	071634	EFT2	

427			
428		:ERROR 215	DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
429	003742 067672	EMT215	
	003744 071422	EHT2	
	003746 071544	EDT2	
	003750 071634	EFT2	
430			
431		:ERROR 216	INCORRECT "IVC" STATUS
432	003752 067704	EMT216	
	003754 071410	EHT1	
	003756 071534	EDT1	
	003760 071624	EFT1	
433			
434		:ERROR 217	INCORRECT "IAE" STATUS
435	003762 067714	EMT217	
	003764 071410	EHT1	
	003766 071534	EDT1	
	003770 071624	EFT1	
436			
437		:ERROR 220	INCORRECT "WLE" STATUS
438	003772 067724	EMT220	
	003774 071410	EHT1	
	003776 071534	EDT1	
	004000 071624	EFT1	
439			
440		:ERROR 221	INCORRECT "DPI" STATUS
441	004002 067734	EMT221	
	004004 071410	EHT1	
	004006 071534	EDT1	
	004010 071624	EFT1	
442			
443		:ERROR 222	RM DID NOT DETECT RMR ERROR
444	004012 067744	EMT222	
	004014 071410	EHT1	
	004016 071534	EDT1	
	004020 071624	EFT1	
445			
446		:ERROR 223	RM DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
447	004022 067754	EMT223	
	004024 071446	EHT223	
	004026 071560	EDT223	
	004030 071650	EFT223	

448				
449			:ERROR	224
450				UNUSED
	004032	000000		0
	004034	000000		0
	004036	000000		0
	004040	000000		0
451				
452			:ERROR	225
453				UNUSED
	004042	000000		0
	004044	000000		0
	004046	000000		0
	004050	000000		0
454				
455			:ERROR	226
456				UNUSED
	004052	000000		0
	004054	000000		0
	004056	000000		0
	004060	000000		0
457				
458			:ERROR	227
459				UNUSED
	004062	000000		0
	004064	000000		0
	004066	000000		0
	004070	000000		0
460				
461			:ERROR	230
462				UNUSED
	004072	000000		0
	004074	000000		0
	004076	000000		0
	004100	000000		0
463				
464			:ERROR	231
465				UNUSED
	004102	000000		0
	004104	000000		0
	004106	000000		0
	004110	000000		0
466				
467			:ERROR	232
468				UNUSED
	004112	000000		0
	004114	000000		0
	004116	000000		0
	004120	000000		0
469				

470			:ERROR	233	UNUSED
471					
	004122	000000		0	
	004124	000000		0	
	004126	000000		0	
	004130	000000		0	
472					
473			:ERROR	234	UNUSED
474					
	004132	000000		0	
	004134	000000		0	
	004136	000000		0	
	004140	000000		0	
475					
476			:ERROR	235	UNUSED
477					
	004142	000000		0	
	004144	000000		0	
	004146	000000		0	
	004150	000000		0	
478					
479			:ERROR	236	UNUSED
480					
	004152	000000		0	
	004154	000000		0	
	004156	000000		0	
	004160	000000		0	
481					
482			:ERROR	237	UNUSED
483					
	004162	000000		0	
	004164	000000		0	
	004166	000000		0	
	004170	000000		0	
484					
485			:ERROR	240	UNUSED
486					
	004172	000000		0	
	004174	000000		0	
	004176	000000		0	
	004200	000000		0	
487					
488			:ERROR	241	UNUSED
489					
	004202	000000		0	
	004204	000000		0	
	004206	000000		0	
	004210	000000		0	
490					
491			:ERROR	242	UNUSED

492
 004212 000000
 004214 000000
 004216 000000
 004220 000000

0
 0
 0
 0

493
 494
 495

;ERROR 243 UNUSED

004222 000000
 004224 000000
 004226 000000
 004230 000000

0
 0
 0
 0

496
 497
 498

;ERROR 244 UNUSED

004232 000000
 004234 000000
 004236 000000
 004240 000000

0
 0
 0
 0

499
 500
 501

;ERROR 245 UNUSED

004242 000000
 004244 000000
 004246 000000
 004250 000000

0
 0
 0
 0

502
 503
 504

;ERROR 246 'ATA' NOT RESET BY GO WHEN 'ERR' 0

004252 070030
 004254 071410
 004256 071534
 004260 071624

EMT246
 EHT1
 EDT1
 EFT1

505
 506
 507

;ERROR 247 'ATA' NOT RESET BY WRITING RMAS

004262 070040
 004264 071410
 004266 071534
 004270 071624

EMT247
 EHT1
 EDT1
 EFT1

508
 509
 510

;ERROR 250 'ATA' WAS RESET BY GO WHEN 'ERR' 1

004272 070052
 004274 071410
 004276 071534
 004300 071624

EMT250
 EHT1
 EDT1
 EFT1

511
 512
 513

;ERROR 251 PROGRAM INTERRUPT WAS NOT GENERATED

	004302	070066		EMT251
	004304	071422		EHT2
	004306	071544		EDT2
	004310	071634		EFT2
514				
515			:ERROR	252 PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
516				
	004312	070074		EMT252
	004314	071422		EHT2
	004316	071544		EDT2
	004320	071634		EFT2
517				
518			:ERROR	253 OFFSET MODE WAS NOT RESET BY WRITING RMDC
519				
	004322	070102		EMT253
	004324	071410		EHT1
	004326	071534		EDT1
	004330	071624		EFT1
520				
521			:ERROR	254 INCORRECT "ILF" STATUS
522				
	004332	070120		EMT254
	004334	071410		EHT1
	004336	071534		EDT1
	004340	071624		EFT1
523				
524			:ERROR	255 INCORRECT "ATA" STATUS
525				
	004342	070130		EMT255
	004344	071410		EHT1
	004346	071534		EDT1
	004350	071624		EFT1
526				
527			:ERROR	256 INCORRECT "ILR" STATUS
528				
	004352	070140		EMT256
	004354	071460		EHT256
	004356	071560		EDT223
	004360	071650		EFT223
529				
530			:ERROR	257 INVALID IAE STATUS DURING SEARCH COMMAND
531				
	004362	070150		EMT257
	004364	071410		EHT1
	004366	071534		EDT1
	004370	071624		EFT1
532				
533			:ERROR	260 "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
534				
	004372	070162		EMT260

Line	Code	Address	Message
	004374	071410	EHT1
	004376	071534	EDT1
	004400	071624	EFT1
535			
536			:ERROR 261 DRIVE EXECUTED SEARCH WITH ERROR SET
537			
	004402	070174	EMT261
	004404	071410	EHT1
	004406	071534	EDT1
	004410	071624	EFT1
538			
539			:ERROR 262 "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
540			
	004412	070214	EMT262
	004414	071410	EHT1
	004416	071534	EDT1
	004420	071624	EFT1
541			
542			:ERROR 263 "SKI" ERROR DURING SEARCH COMMAND
543			
	004422	070224	EMT263
	004424	071410	EHT1
	004426	071534	EDT1
	004430	071624	EFT1
544			
545			:ERROR 264 "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
546			
	004432	070234	EMT264
	004434	071410	EHT1
	004436	071534	EDT1
	004440	071624	EFT1
547			
548			:ERROR 265 ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
549			
	004442	070254	EMT265
	004444	071410	EHT1
	004446	071534	EDT1
	004450	071624	EFT1
550			
551			:ERROR 266 DEVICE FAULT (DVC) DURING SEARCH
552			
	004452	070274	EMT266
	004454	071410	EHT1
	004456	071534	EDT1
	004460	071624	EFT1
553			
554			:ERROR 267 SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
555			ADDRESS IS TOO LARGE
556			
	004462	070306	EMT267

	004464	071410	EHT1	
	004466	071534	EDT1	
	004470	071624	EFT1	
557				
558				:ERROR 270 OPI ERROR DURING SEARCH BECAUSE MOL = 0
559				
	004472	070324	EMT270	
	004474	071410	EHT1	
	004476	071534	EDT1	
	004500	071624	EFT1	
560				
561				:ERROR 271 OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
562				: DIDN'T DROP
563				
	004502	070340	EMT271	
	004504	071410	EHT1	
	004506	071534	EDT1	
	004510	071624	EFT1	
564				
565				:ERROR 272 LOST MOL DURING SEARCH, OPI IS NOT SET
566				
	004512	070356	EMT272	
	004514	071410	EHT1	
	004516	071534	EDT1	
	004520	071624	EFT1	
567				
568				:ERROR 273 PIP STIL SET AFTER SEARCH
569				
	004522	070374	EMT273	
	004524	071410	EHT1	
	004526	071534	EDT1	
	004530	071624	EFT1	
570				
571				:ERROR 274 PARITY ERROR OCCURRED WHILE WRITING REMOTE
572				: REGISTERS BUT MXF DID NOT SET
573				
	004532	070412	EMT274	
	004534	071410	EHT1	
	004536	071534	EDT1	
	004540	071624	EFT1	
574				
575				:ERROR 275 MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
576				: COMMAND STARTED
577				
	004542	070430	EMT275	
	004544	071410	EHT1	
	004546	071534	EDT1	
	004550	071624	EFT1	
578				
579				:ERROR 276 'OPI' ERROR DURING DATA TRANSFER BECAUSE 'MOL' WAS

580
581

004552 070442
004554 071410
004556 071534
004560 071624

ZERO
EMT276
EHT1
EDT1
EFT1

582
583
584
585
586

004562 070456
004564 071410
004566 071534
004570 071624

:ERROR 277
:
:

'DPI' ERROR DURING DATA TRANSFER BECAUSE 1) ON
CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
3) RUN TIMED OUT

EMT277
EHT1
EDT1
EFT1

587
588
589
590

004572 070474
004574 071410
004576 071534
004600 071624

:ERROR 300
:

'IVC' ERROR DURING DATA TRANSFER BECAUSE VOLUME
WAS NOT VALID

EMT300
EHT1
EDT1
EFT1

591
592
593
594

004602 070514
004604 071410
004606 071534
004610 071624

:ERROR 301
:

ERRONEOUS 'IVC' ERROR DURING DATA TRANSFER - VOLUME
IS VALID

EMT301
EHT1
EDT1
EFT1

595
596
597

004612 070536
004614 071410
004616 071534
004620 071624

:ERROR 302

'ILR' ERROR DURING DATA TRANSFER

EMT302
EHT1
EDT1
EFT1

598
599
600

004622 070550
004624 071410
004626 071534
004630 071624

:ERROR 303

'ILF' ERROR DURING DATA TRANSFER

EMT303
EHT1
EDT1
EFT1

601
602
603

004632 070562
004634 071410
004636 071534

:ERROR 304

'RMR' ERROR DURING DATA TRANSFER

EMT304
EHT1
EDT1

004640	071624	EFT1	
604			
605		:ERROR	305 INCORRECT "IAE" STATUS DURING DATA TRANSFER
606			
004642	070574	EMT305	
004644	071410	EHT1	
004646	071534	EDT1	
004650	071624	EFT1	
607			
608		:ERROR	306 "SKI" ERROR DURING DATA TRANSFER
609			
004652	070606	EMT306	
004654	071410	EHT1	
004656	071534	EDT1	
004660	071624	EFT1	
610			
611		:ERROR	307 DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
612			
004662	070616	EMT307	
004664	071410	EHT1	
004666	071534	EDT1	
004670	071624	EFT1	
613			
614		:ERROR	310 DEVICE FAULT DURING DATA TRANSFER
615			
004672	070636	EMT310	
004674	071410	EHT1	
004676	071534	EDT1	
004700	071624	EFT1	
616			
617		:ERROR	311 LOSS OF BIT CLOCK DURING DATA TRANSFER
618			
004702	070650	EMT311	
004704	071410	EHT1	
004706	071534	EDT1	
004710	071624	EFT1	
619			
620		:ERROR	312 LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
621			
004712	070662	EMT312	
004714	071410	EHT1	
004716	071534	EDT1	
004720	071624	EFT1	
622			
623		:ERROR	313 UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
624			
004722	070674	EMT313	
004724	071410	EHT1	
004726	071534	EDT1	
004730	071624	EFT1	

625			
626		:ERROR 314	DRIVE TIMING ERROR DURING DATA TRANSFER
627	004732 070714	EMT314	
	004734 071410	EHT1	
	004736 071534	EDT1	
	004740 071624	EFT1	
628			
629		:ERROR 315	WRITE LOCK ERROR
630	004742 070726	EMT315	
	004744 071410	EHT1	
	004746 071534	EDT1	
	004750 071624	EFT1	
631			
632		:ERROR 316	ERRONEOUS WRITE LOCK ERROR
633	004752 070740	EMT316	
	004754 071410	EHT1	
	004756 071534	EDT1	
	004760 071624	EFT1	
634			
635		:ERROR 317	HEADER CRC ERROR DURING DATA TRANSFER
636	004762 070752	EMT317	
	004764 071410	EHT1	
	004766 071534	EDT1	
	004770 071624	EFT1	
637			
638		:ERROR 320	FORMAT ERROR DURING DATA TRANSFER
639	004772 070762	EMT320	
	004774 071410	EHT1	
	004776 071534	EDT1	
	005000 071624	EFT1	
640			
641		:ERROR 321	HEADER COMPARE ERROR DURING DATA TRANSFER
642	005002 070772	EMT321	
	005004 071410	EHT1	
	005006 071534	EDT1	
	005010 071624	EFT1	
643			
644		:ERROR 322	HEADER ERRORS SHOULD NOT BE SET
645	005012 071002	EMT322	
	005014 071410	EHT1	
	005016 071534	EDT1	
	005020 071624	EFT1	

646
647
648

005022 071010
005024 071410
005026 071534
005030 071624

:ERROR 323 DATA CHECK ERROR DURING DATA TRANSFER
EMT323
EHT1
EDT1
EFT1

649
650
651

005032 071020
005034 071410
005036 071534
005040 071624

:ERROR 324 CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
EMT324
EHT1
EDT1
EFT1

652
653
654

005042 071032
005044 071410
005046 071534
005050 071624

:ERROR 325 UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
EMT325
EHT1
EDT1
EFT1

655
656
657

005052 071044
005054 071410
005056 071534
005060 071624

:ERROR 326 DATA PARITY ERROR DURING READ COMMAND
EMT326
EHT1
EDT1
EFT1

658
659
660

005062 071062
005064 071410
005066 071534
005070 071624

:ERROR 327 OFFSET MODE NOT RESET BY WRITE COMMAND
EMT327
EHT1
EDT1
EFT1

661
662
663

005072 071074
005074 071410
005076 071534
005100 071624

:ERROR 330 DATA PARITY ERROR DURING WRITE COMMAND
EMT330
EHT1
EDT1
EFT1

664
665
666

005102 071104
005104 071410
005106 071534
005110 071624

:ERROR 331 WRITE CLOCK FAILURE DURING WRITE COMMAND
EMT331
EHT1
EDT1
EFT1

667

668			:ERROR 332	DATA LATE ERROR DURING DATA TRANSFER
669	005112	071116		
	005114	071410	EMT332	
	005116	071534	EHT1	
	005120	071624	EDT1	
			EFT1	
670				
671			:ERROR 333	PIP STIL SET AFTER DATA TRANSFER - SKI = 0
672	005122	071130		
	005124	071410	EMT333	
	005126	071534	EHT1	
	005130	071624	EDT1	
			EFT1	
673				
674			:ERROR 334	LOST MOL DURING DATA TRANSFER - OP! = 0
675	005132	071146		
	005134	071410	EMT334	
	005136	071534	EHT1	
	005140	071624	EDT1	
			EFT1	
676				
677			:ERROR 335	LOST VOLUME VALID DURING DATA TRANSFER - IVC 0
678	005142	071164		
	005144	071410	EMT335	
	005146	071534	EHT1	
	005150	071624	EDT1	
			EFT1	
679				
680			:ERROR 336	DATA READ DOES NOT COMPARE WITH DATA WRITTEN
681	005152	071204		
	005154	071472	EMT336	
	005156	071570	EHT336	
	005160	071660	EDT336	
			EFT336	
682				
683			:ERROR 337	WRITE CHECK ERROR NOT DETECTED
684	005162	071214		
	005164	071504	EMT337	
	005166	071600	EHT337	
	005170	071670	EDT337	
			EFT337	
685				
686			:ERROR 340	WRITE CHECK ERROR AT UNEXPECTED ADDRESS
687	005172	071224		
	005174	071472	EMT340	
	005176	071570	EHT336	
	005200	071660	EDT336	
			EFT336	
688				
689			:ERROR 341	INCORRECT DATA DURING WRITE CHECK ERROR

ERROR POINTER TABLE

690	005202 071236	EMT341	
	005204 071472	EHT336	
	005206 071570	EDT336	
	005210 071660	EFT336	
691			
692		:ERROR 342	"IVC" ERROR NOT DETECTED DURING DATA TRANSFER
693	005212 071244	EMT342	
	005214 071410	EHT1	
	005216 071534	EDT1	
	005220 071624	EFT1	
694			
695		:ERROR 343	"FER" NOT DETECTED DURING DATA TRANSFER
696	005222 071256	EMT343	
	005224 071410	EHT1	
	005226 071534	EDT1	
	005230 071624	EFT1	
697			
698		:ERROR 344	"HCE" NOT DETECTED DURING DATA TRANSFER
699	005232 071270	EMT344	
	005234 071516	EHT344	
	005236 071610	EDT344	
	005240 071700	EFT344	
700			
701		:ERROR 345	"BSE" NOT DETECTED DURING DATA TRANSFER
702	005242 071302	EMT345	
	005244 071410	EHT1	
	005246 071534	EDT1	
	005250 071624	EFT1	
703			
704		:ERROR 346	HEADER ERROR WAS DETECTED W/ HCI SET
705	005252 071312	EMT346	
	005254 071410	EHT1	
	005256 071534	EDT1	
	005260 071624	EFT1	
706			
707		:ERROR 347	DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
708	005262 071326	EMT347	
	005264 071410	EHT1	
	005266 071534	EDT1	
	005270 071624	EFT1	
709			
710		:ERROR 350	LOST VOLUME VALID DURING SEARCH - "IVC" - 0
711			

005272	071340	EMT350
005274	071410	EHT1
005276	071534	EDT1
005300	071624	EFT1
712		
713		:ERROR 351 'ATA' DID NOT SET DURING SEARCH
714		
005302	071356	EMT351
005304	071410	EHT1
005306	071534	EDT1
005310	071624	EFT1
715		
716		:ERROR 352 PROGRAM TIMEOUT WHILE TESTING RMLA
717		
005312	071366	EMT352
005314	000000	0
005316	000000	0
005320	000000	0
718		
719		:ERROR 353 LOOK AHEAD TEST FAILS
720		
005322	071372	EMT353
005324	071530	EHT353
005326	071622	EDT353
005330	071710	EFT353
721		
722		:ERROR 354 BSE SHOULD NOT BE SET
723		
005332	071402	EMT354
005334	071410	EHT1
005336	071534	EDT1
005340	071624	EFT1
724		
725		:PUT ERROR TABLE HERE

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

```
.SBTTL  ERROR TABLE USAGE

:THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
:NUMBER, I.E.,
:
:      EMT - ERROR MESSAGE TABLE ADDRESS
:      EHT - ERROR HEADER TABLE ADDRESS
:      EDT - ERROR DATA TABLE ADDRESS
:      EFT - ERROR FORMAT TABLE ADDRESS
:
:THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
:FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
:OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
:TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
:NO MESSAGE TO BE TYPED FOR THE ERROR.
:
:SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
:OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
:IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
:DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
:HEADER.  THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
:BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
:MUST ALSO HAVE A FORMAT.
:
:IN SUMMARY,
:      EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
:      EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
:      OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
```



```

1      ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS
2
3 005342 011600      BADTMO: MOV      (SP),R0      ;SAVE PC WHERE THE TIME OUT OCCURED
4 005344 005740      TST      -(R0)      ;ADJUST PC -2
5 005346 022626      CMP      (SP)+,(SP)+    ;RESTORE STACK POINTER
6 005350 104401 005356  TYPE     .65$      ;:TYPE ASCIZ STRING
   005354 000417      BR       64$      ;:GET OVER THE ASCIZ
   ;:65$: .ASCIZ <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
   64$:
7 005414 010046      MOV      R0,-(SP)      ;SETUP FOR TYPING OUT PC
8 005416 104402      TYPOC
9 005420 000240      NOP
   ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
   ;TO STOP ON UNEXPECTED TIMEOUT.
10
11
12      .SBTTL START OF PROGRAM
13
14 005422 012737 177777 001330 START1: MOV      #-1,CHGADR    ;CHANGE RH/RM BUS ADDRESS
15 005430 000402      BR       START2
16
17 005432 005037 001330 START:  CLR      CHGADR      ;NO CHANGE IN ADDRESS
18 005436 000240 START2: NOP
19 005440 005227 000000      INC      #0      ;TTY LOOP, WAIT FOR INCREMENT
20 005444 001375      BNE     .-4      ;OF WORD
21 005446 000005      RESET     ;RESET THE WORLD
22
23      .SBTTL INITIALIZE THE COMMON TAGS
   ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
   MOV      #$CMTAG,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      #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
   MOV      #340,@#PWRVEC+2 ;;LEVEL 7
   MOV      $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
   CLR      $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
   CLR      $ESCAPE     ;;CLEAR THE ESCAPE ON ERROR ADDRESS
   MOV     #1,$ERMAX    ;;ALLOW ONE ERROR PER TEST
   MOV     #,$SLPADR    ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
   MOV     #,$SLPERR    ;;SETUP THE ERROR LOOP ADDRESS
   ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
   ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
   MOV     @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
   MOV     #64$,@#ERRVEC ;;SET UP ERROR VECTOR
   MOV     #DSWR,SWR    ;;SETUP FOR A HARDWARE SWICH REGISTER
   MOV     #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
   CMP     #-1,@SWR    ;;TRY TO REFERENCE HARDWARE SWR
   BNE     66$        ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
   ;;AND THE HARDWARE SWR IS NOT -1
   BR     65$        ;;BRANCH IF NO TIMEOUT

```

```

005650 012716 005656      64$:  MOV    #65$, (SP)      ;;SET UP FOR TRAP RETURN
005654 000002
005656 012737 000176 001154 65$:  MOV    #SWREG,SWR      ;;POINT TO SOFTWARE SWR
005664 012737 000174 001156      MOV    #DISPREG,DISPLAY
005672 012637 000004      66$:  MOV    (SP)+,@#ERRVEC    ;;RESTORE ERROR VECTOR

005676 005037 001230      CLR    $PASS              ;;CLEAR PASS COUNT
005702 132737 000200 001243      BITB  #APTSIZE,$ENVM      ;;TEST USER SIZE UNDER APT
005710 001403      BEQ    67$                ;;YES,USE NON-APT SWITCH
005712 012737 001244 001154      MOV    #$$SWREG,SWR      ;;NO,USE APT SWITCH REGISTER
005720
24      ;SETUP 'TIMEOUT' TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
25 005720 012737 005342 000004      MOV    #BADTMO,ERRVEC    ;;SETUP FOR UNEXPECTED TIMEOUT
26 005726 012737 000300 000006      MOV    #PR6,ERRVEC+2    ;;LEVEL 6
27
28      .SBTTL  TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005734 005227 177777      INC    #-1                ;;FIRST TIME?
005740 001035      BNE    68$                ;;BRANCH IF NO
005742 022737 032122 000042      CMP    #SENDAD,@#42      ;;ACT-11?
005750 001431      BEQ    68$                ;;BRANCH IF YES
005752 104401 005760      TYPE  ,69$                ;;TYPE ASCIZ STRING
005756 000426      BR     68$                ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ <CRLF>@CZRMNBO - RM05/3/2 FUNCTIONAL TEST, PT 2@<CRLF>
      68$:
      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
006034 005737 000042      TST    @#42                ;;ARE WE RUNNING UNDER XXDP/ACT?
006040 001012      BNE    70$                ;;BRANCH IF YES
006042 123727 001242 000001      CMPB  $ENV,#1            ;;ARE WE RUNNING UNDER APT?
006050 001406      BEQ    70$                ;;BRANCH IF YES
006052 023727 001154 000176      CMP    SWR,#SWREG        ;;SOFTWARE SWITCH REG SELECTED?
006060 001005      BNE    71$                ;;BRANCH IF NO
006062 104407      GTSWR                      ;;GET SOFT-SWR SETTINGS
006064 000403
006066 112737 000001 001150 70$:  MOVB  #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
006074      71$:

29
30      ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
31      ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
32
33 006074 005037 001332      CLR    XXDP                ;CLEAR 'XXDP' LOAD DEVICE STORAGE
34 006100 122737 000016 000041      CMPB  #16,@#41            ;LOADED FROM AN RM05/3/2 ?
35 006106 001160      BNE    3$                 ;BRANCH IF NOT
36 006110 013737 000040 001332      MOV    @#40,XXDP          ;GET DEVICE INDICATOR AND NUMBER
37 006116 122737 000007 001332      CMPB  #7,XXDP            ;IS IT A VALID NUMBER ?
38 006124 103002      BHIS  1$                 ;YES
39 006126 105037 001332      CLRB  XXDP                ;NO, DEFAULT TO DRIVE 0
40 006132 005737 000042      1$:  TST    @#42                ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
41 006136 001425      BEQ    2$                 ;BR IF NEITHER
42 006140 104401 006146      TYPE  ,73$                ;;TYPE ASCIZ STRING
006144 000412      BR     72$                ;;GET OVER THE ASCIZ
      ;;73$: .ASCIZ <CRLF>/NOT TESTING DRIVE /
      72$:
43 006172 005046      CLR    -(SP)              ;CLEAR WORD ON STACK
44 006174 113716 001332      MOVB  XXDP,(SP)          ;GET DRIVE ADDRESS
45 006200 104403      TYPOS                      ;TYPE THE ADDRESS
46 006202 001      .BYTE 1                  ;ONLY 1 CHARACTER

```

```

47 006203      000      .BYTE 0      ;SUPRESS LEADING ZEROS
48 006204 104401 001217  TYPE $CRLF ;CR-LF
49 006210 000517      BR 3$      ;GET NUMBER OF DRIVES
50
51 006212 005227 177777 2$: INC #-1 ;FIRST TIME THRU HERE ?
52 006216 001114      BNE 3$      ;NO
53 006220 104401 006226  TYPE .75$ ;TYPE ASCIZ STRING
    006224 000410      BR .74$ ;GET OVER THE ASCIZ
    ;:75$: .ASCIZ <CRLF>/TO TEST DRIVE /
    ;:74$:
54 006246 005046      CLR -(SP) ;CLEAR WORD ON STACK
55 006250 113716 001332  MOVB XXDP,(SP) ;GET DRIVE ADDRESS
56 006254 104403      TYPOS ;TYPE DRIVE ADDRESS
57 006256      001      .BYTE 1 ;ONLY 1 CHARACTER
58 006257      000      .BYTE 0 ;SUPRESS LEADING ZEROS
59 006260 104401 006266  TYPE .77$ ;TYPE ASCIZ STRING
    006264 000431      BR .76$ ;GET OVER THE ASCIZ
    ;:77$: .ASCIZ /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
    ;:76$:
60 006350 104401 006356  TYPE .78$ ;TYPE ASCIZ STRING
    006354 000435      BR 3$ ;GET OVER THE ASCIZ
    ;:78$: .ASCIZ /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
    ;:3$:
64 ;CHECK FOR AUTO MODE OR STANDLONE MODE
65 006450 005737 000042  TST @#42 ;RUNNING IN AUTO MODE ?
66 006454 001561      BEQ STANDALONE ;BR IF NO
67 006456 012737 000377 001300  MOV #377,$DEV ;SET DEVICE MAP FOR ALL DRIVES
68
69 ;PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
70 XSIZ:
71 006464 132737 000200 001243  BITB #BIT7,$ENVM ;SIZING ALLOWED ?
72 006472 001146      BNE 12$ ;NO
73
74 006474 005001      CLR R1 ;START FROM DRIVE 0
75 006476 013700 001276  MOV $BASE,R0 ;LOAD THE BASE ADDRESS
76 006502 104401 064350  TYPE ,SYSTAT ;TYPE 'UNIT STATUS:'
77
78 006506 136137 064666 001300 1$: BITB ATNTBL(R1),$DEV ;IS DEVICE PRESENT IN MAP ?
79 006514 001531      BEQ 11$ ;BR IF NO
80 006516 104401 001217  TYPE $CRLF ;CR-LF
81 006522 010146      MOV R1,-(SP) ;SAVE R1 FOR TYPEOUT
    006524 104403      TYPOS ;GO TYPE--OCTAL ASCII
    006526      002      .BYTE 2 ;TYPE 2 DIGIT(S)
    006527      000      .BYTE 0 ;SUPPRESS LEADING ZEROS
82 006530 104401 064560  TYPE ,BLNKS4 ;TYPE 4 BLANKS
83
84 006534 012760 000040 000010  MOV #CLR, RMCS2(R0) ;CLEAR MASS BUS
85 006542 010160 000010  MOV R1, RMCS2(R0) ;LOAD THE DRIVE ADDRESS
86 006546 005760 000012  TST RMD5(R0) ;ACCESS DRIVE REGISTER
87 006552 032760 010000 000010  BIT #NED, RMCS2(R0) ;IS DRIVE PRESENT ?
88 006560 001051      BNE 3$ ;BR IF NO
89 006562 032760 004000 000000  BIT #DVA, RMCS1(R0) ;IS DRIVE AVAILABLE ?
90 006570 001450      BEQ 4$ ;BR IF NO
91 006572 016002 000026  MOV RMDT(R0),R2 ;SAVE DRIVE TYPE REGISTER IN R2
92 006576 012737 064366 006776  MOV #SRM02,10$ ;ASSUME RM02 DEVICE
93 006604 022702 020025  CMP #20025,R2 ;SINGLE PORT RM02 ?
94 006610 001430      BEQ 2$ ;BR IF YES
    
```

95	006612	022702	024025			CMP	#24025,R2	:DUAL PORT RM02 ?
96	006616	001425				BEQ	2\$:BR IF YES
97	006620	012737	064373	006776		MOV	#\$RM03,10\$:ASSUME RM03 DEVICE
98	006626	022702	020024			CMP	#20024,R2	:SINGLE PORT RM03 ?
99	006632	001417				BEQ	2\$:BR IF YES
100	006634	022702	024024			CMP	#24024,R2	:DUAL PORT RM03 ?
101	006640	001414				BEQ	2\$:BR IF YES
102	006642	012737	064400	006776		MOV	#\$RM05,10\$:ASSUME RM05 DEVICE
103	006650	022702	020027			CMP	#20027,R2	:SINGLE PORT RM05 ?
104	006654	001406				BEQ	2\$:BR IF YES
105	006656	022702	024027			CMP	#24027,R2	:DUAL PORT RM05 ?
106	006662	001403				BEQ	2\$:BR IF YES
107	006664	104401	064405			TYPE	,NOTRM	:DRIVE NOT AN RM05/3/2
108	006670	000412				BR	5\$:CHECK NEXT DRIVE
109	006672	032760	010000	000012	2\$:	BIT	#MOL,RMDS(R0)	:IS MEDIUM ON LINE ?
110	006700	001412				BEQ	6\$:BR IF NO
111	006702	000417				BR	7\$	
112								
113	006704	104401	00443		3\$:	TYPE	,NOTPRS	:DRIVE NOT PRESENT
114	006710	000402				BR	5\$:CHECK NEXT DRIVE
115								
116	006712	104401	064460		4\$:	TYPE	,NOTAVL	:DRIVE NOT AVAILABLE
117	006716	146137	064666	001300	5\$:	BICB	ATNTBL(R1),\$DEV	:CLEAR DEVICE FROM BIT MAP
118	006724	000425				BR	11\$:CHECK NEXT DRIVE
119								
120	006726	104401	064477		6\$:	TYPE	,UNTOFF	:DRIVE OFFLINE
121	006732	146137	064666	001300		BICB	ATNTBL(R1),\$DEV	:CLEAR DEVICE FROM BIT MAP
122	006740	000413				BR	9\$:PRINT DRIVE TYPE
123								
124	006742	005737	001332		7\$:	TST	XXDP	:LOADED FROM RM05/3/2 ?
125	006746	001406				BEQ	8\$:NO
126	006750	123701	001332			CMPB	XXDP,R1	:IS THIS THE DRIVE ?
127	006754	001360				BNE	5\$:BR IF NO
128	006756	104401	064426			TYPE	,LODEV	:DRIVE IS LOAD DEVICE
129	006762	000755				BR	5\$	
130								
131	006764	104401	064510		8\$:	TYPE	,UNTON	:DRIVE ONLINE
132	006770	104401	064562		9\$:	TYPE	,BLNKS2	:TYPE 2 BLANKS
133	006774	104401				TYPE		:PRINT DRIVE TYPE
134	006776	000000			10\$:	.WORD	0	:MESSAGE ADDRESS HERE
135								
136	007000	005201			11\$:	INC	R1	:INCREMENT THE DRIVE ADDRESS
137	007002	020127	000007			CMP	R1,#7	:ALL DRIVES ARE CHECKED ?
138	007006	003637				BLE	1\$:BRANCH IF NOT
139								
140	007010	104401	001217		12\$:	TYPE	,\$CRLF	:CR-LF
141	007014	000137	007476			JMP	CMNSTART	:JUMP TO COMMON START

```

1
2
3 007020
4 007020 004737 061230
5
6 007024 005227 177777
7 007030 001023
8
9
10
11 007032 104401 063450
12 007036 104411
13 007040 012637 001176
14 007044 123727 001176 000131
15 007052 001005
16 007054 104401 001176
17 007060 104401 101360
18 007064 000414
19 007066 104401 064554
20 007072 104401 001217
21 007076 000407
22
23
24 007100
25 007100 005737 001330
26 007104 001457
27 007106 005037 001330
28 007112 104401 001217
29
30
31 007116
32 007116 104401 064025
33 007122 013746 001276
    007126 104402
34 007130 104401 064562
35 007134 104413
36 007136 012637 001176
37 007142 001412
38 007144 022737 160000 001176
39 007152 101403
40 007154 104401 064035
41 007160 000756
42 007162 013737 001176 001276
43
44 007170 104401 064077
45 007174 005046
46 007176 113716 001272
47 007202 104402
48 007204 104401 064562
49 007210 104413
50 007212 012637 001176
51 007216 001412
52 007220 022737 001000 001176
53 007226 101003
54 007230 104401 064106
55 007234 000755
56 007236 113737 001176 001272

```

.SBTTL STANDALONE INPUT ROUTINES

STANDALONE:

```

JSR PC,$TKINT ;INITIALIZE CONSOLE
INC #-1 ;FIRST TIME THRU HERE ?
BNE 2$ ;BR IF NO

```

;SEE IF OPERATOR WANTS HELP TEXT

```

TYPE ,MSHELP ;WANT HELP ?
RDCHR ;GET RESPONSE
MOV (SP)+,$TMP1 ;SAVE AND ECHO RESPONSE
CMPB $TMP1,#'Y ;WAS IT A YES RESPONSE ?
BNE 1$ ;NO
TYPE , $TMP1 ;TYPE 'Y'
TYPE ,HELP ;YES - TYPE HELP TEXT
BR 3$
1$: TYPE ,N ;TYPE 'N'
TYPE , $CRLF ;CR-LF
BR 3$

```

;SEE IF USER WANTS TO CHANGE UNIBUS ADDRESS

```

2$: TST CHGADR ;CHANGE RH/RM BUS ADDRESS ?
BEQ 7$ ;BR IF NO
CLR CHGADR ;NO CHANGE NEXT TIME
TYPE , $CRLF ;CR-LF

```

;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY

```

3$: TYPE ,CNSL01 ;TYPE CURRENT BUS ADDRESS
MOV $BASE,-(SP) ;SAVE $BASE FOR TYPEOUT
TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE ,BLNKS2 ;TYPE 2 BLANKS
RDOCT ;GET NEW BUS ADDRESS
MOV (SP)+,$TMP1 ;CARRIAGE RETURN ?
BEQ 5$ ;YES-SKIP TO NEXT ENTRY
CMP #160000,$TMP1 ;BASE ADDRESS IN I/O PAGE ?
BLOS 4$ ;YES
TYPE ,CNSL02 ;TYPE WARNING MESSAGE
BR 3$ ;TRY AGAIN
MOV $TMP1,$BASE ;STORE NEW BUS ADDRESS
5$: TYPE ,CNSL03
CLR -(SP)
MOVB $VECT1,(SP) ;GET CURRENT VECTOR ADDRESS
TYPOC ;TYPE 2 BLANKS
RDOCT ;GET NEW VECTOR ADDRESS
MOV (SP)+,$TMP1 ;CARRIAGE RETURN?
BEQ 7$ ;YES-SKIP TO NEXT ENTRY
CMP #1000,$TMP1 ;VECTOR ADDRESS < 1000 ?
BHI 6$ ;YES.!
TYPE ,CNSL04 ;TYPE WARNING MESSAGE
BR 5$ ;RETRY
MOVB $TMP1,$VECT1 ;STORE NEW VECTOR ADDRESS
6$:

```

```

57
58
59 007244 005227 177777
60 007250 001002
61 007252 104401 064142
62 007256 104401 001217
63 007262 005037 001300
64 007266 104401 064326
65 007272 104411
66 007274 012637 001176
67 007300 023727 001176 000101
68 007306 001007
69 007310 104401 063434
70 007314 012737 000377 001300
71 007322 000137 006464
72
73 007326 023727 001176 000015 10$:
74 007334 001436
75 007336 104401 001176
76 007342 023727 001176 000060
77 007350 002430
78 007352 023727 001176 000067
79 007360 003427
80 007362 000423
81
82 007364 104411
83 007366 012637 001176
84 007372 023727 001176 000015
85 007400 001432
86 007402 104401 063445
87 007406 104401 001176
88 007412 023727 001176 000060
89 007420 002404
90 007422 023727 001176 000067
91 007430 003403
92 007432 104401 064304
93 007436 000711
94
95 007440 013701 001176
96 007444 042701 177770
97 007450 156137 064666 001300
98 007456 122737 000377 001300
99 007464 101337
100 007466 104401 001217
101 007472 000137 006464

; DIALOGUE TO INPUT DEVICE NUMBERS
7$: INC # - 1 ; FIRST TIME THRU ?
    BNE 8$ ; BR IF NO
    TYPE , CNSL07 ; TYPE INPUT INSTRUCTIONS
8$: TYPE , $CRLF ; CR-LF
9$: CLR $DEV ; CLEAR DEVICE MAP
    TYPE , MSDRVS ; TYPE 'DRIVE(S): '
    RDCHR
    MOV (SP)+, $TMP1 ; GET RESPONSE
    CMP $TMP1, #'A ; IS INPUT 'A' ?
    BNE 10$ ; NO
    TYPE , ALL ; YES, TYPE 'ALL' AND GO
    MOV #377, $DEV ; SET DEVICE MAP FOR ALL DRIVES
    JMP XSIZ ; AUTO SIZE.

10$: CMP $TMP1, #CR ; CARRIAGE RETURN ?
    BEQ 12$ ; YES
    TYPE , $TMP1 ; ECHO RESPONSE
    CMP $TMP1, #'0 ; NUMBER < 0 ?
    BLT 12$ ; YES
    CMP $TMP1, #'7 ; NUMBER > 7 ?
    BLE 13$ ; NO
    BR 12$ ; ILLEGAL INPUT

11$: RDCHR
    MOV (SP)+, $TMP1 ; GET RESPONSE
    CMP $TMP1, #CR ; CARRIAGE RETURN ?
    BEQ 14$ ; YES
    TYPE , COMMA ; TYPE ', '
    TYPE , $TMP1 ; ECHO RESPONSE
    CMP $TMP1, #'0 ; NUMBER < 0 ?
    BLT 12$ ; YES
    CMP $TMP1, #'7 ; NUMBER > 7 ?
    BLE 13$ ; NO
    TYPE , CNSL08 ; TYPE CR-LF '' ? ILLEGAL INPUT''
    BR 9$ ; RETRY

12$:
13$: MOV $TMP1, R1 ; R1 - DRIVE NUMBER
    BIC #^C7, R1
    BISB ATNTBL(R1), $DEV ; SET DEVICE IN MAP
    CMPB #377, $DEV ; DONE ?
    BHI 11$ ; NO
14$: TYPE , $CRLF ; CR-LF
    JMP XSIZ ; GO SIZE DEVICES

```

```

1
2 007476          :ASSEMBLE TEST QUE FROM DEVICE MAP
3 007476 104401 064520 CMNSTART:
4 007502 013700 001300      TYPE      ,DRIVES      :TYPE 'DRIVE(S) TO BE TESTED'
5 007506 001004          MOV      $DEVN,R0      :R0 = DEVICE MAP
6 007510 104401 063445      BNE      1$          :BR IF DRIVES TO TEST
7 007514 104401 064547      TYPE      ,COMMA      :TYPE ' '
8 007520 012701 001470      TYPE      ,NONE      :TYPE 'NONE'
9 007524 010137 001466      1$: MOV      #TSTQUE+2,R1 :R1 = ADDRESS OF FIRST ENTRY IN QUE
10 007530 012702 000001      MOV      R1,TSTQUE   :INITIALIZE ENTRY POINTER
11 007534 005003          MOV      #1,R2       :R2 = DEVICE POINTER
12 007536 030200          CLR      R3          :R3 = DEVICE NUMBER
13 007540 001413          2$: BIT      R2,R0    :IS THIS DEVICE IN MAP ?
14 007542 104401 063445      BEQ      3$          :NO
15 007546 010311          TYPE      ,COMMA      :TYPE ' '
16 007550 010346          MOV      R3,(R1)     :YES - ENTER DEVICE NUMBER IN QUE
    007552 104403          MOV      R3,-(SP)    :SAVE R3 FOR TYPEOUT
    007554 001          TYPOS      1        :GO TYPE--OCTAL ASCII
    007555 000          .BYTE      0        :TYPE 1 DIGIT(S)
17 007556 116361 064666 000001 MOVVB   ATNTBL(R3),1(R1) :SUPPRESS LEADING ZEROS
18 007564 062701 000002      ADD      #2,R1      :ENTER ATTENTION BIT IN QUE
19 007570 006302          3$: ASL      R2        :ADVANCE ENTRY POINTER
20 007572 105702          TSTB     R2         :ADVANCE DEVICE POINTER
21 007574 001402          BEQ      4$          :DONE ALL DEVICES ?
22 007576 005203          INC      R3         :YES
23 007600 000756          BR       2$          :ADVANCE DEVICE NUMBER
24 007602 005011          4$: CLR      (R1)     :ENTER NEXT DEVICE
25 007604 104401 001217      TYPE      ,$CRLF     :TERMINATE TEST QUE
26
27
28 007610 004737 036460      :SIZE FOR CLOCK
29 007614 000425          JSR      PC,SIZCLK   :SEE IF CLOCK PRESENT
30 007616 104401 007624      BR       6$          :YES - CLOCK IS PRESENT
    007622 000413          TYPE      ,65$      :TYPE ASCII STRING
    007652          BR       64$     :GET OVER THE ASCIIZ
    64$: .ASCIIZ <CRLF>/NO 'L' OR 'P' CLOCK/
31 007652 005737 000042      TST      @#42        :ANY MONITOR PRESENT ?
32 007656 001002          BNE      5$          :BR IF YES
33 007660 000137 005432      JMP      START      :JUMP TO START
34 007664 000137 032112      5$: JMP      $GET42    :RETURN CONTROL TO MONITOR
35 007670
36
37 007670 000240          READY: NOP          :READY TO START TEST
38 007672 105737 001300      TSTB     $DEVN      :ANY DRIVES IN MAP ?
39 007676 001007          BNE      2$          :BR IF YES
40 007700 005737 000042      TST      @#42        :ANY MONITOR PRESENT ?
41 007704 001002          BNE      1$          :BR IF YES
42 007706 000137 005432      JMP      START      :JUMP TO START
43 007712 000137 032112      1$: JMP      $GET42    :RETURN CONTROL TO MONITOR
44
45 007716 105037 001116      2$: CLRB     $STNM     :RESET TEST NUMBER
46 007722 005037 001206      CLR      $TIMES     :INITIALIZE NUMBER OF ITERATIONS
47 007726 005037 001326      CLR      CTLFG      :CLEAR CONTROL-C FLAG
48 007732 004737 061230      JSR      PC,$TKINT   :INITIALIZE ITY
49 007736 012746 000240      MOV      #PR5,-(SP)  :PUT NEW PS ON STACK
    007742 012746 007750      MOV      #64$,-(SP)  :PUT NEW PC ON STACK
    007746 000002      RTI                :POP NEW PC AND PS

```

```

007750
50 007750 117737 171512 001234 64$:      MOVB  @TSTQUE,$UNIT  ;LOAD DRIVE NUMBER
51 007756 005037 001512          CLR   MEDENB        ;CLEAR MEDIA ENABLE
52
53          ;CLEAR MASSBUS CONTROLLER, SELECT DRIVE AND DETERMINE THE LAST TRACK
54          ;OF THE DIFFERENT DRIVE TYPES
55
56 007762 012737 002000 001334      MOV   #TA4,LSTRK    ;ASSUME LAST TRACK FOR RM02/3 - 4.
57 007770 013700 001276          MOV   $BASE,R0      ;R0 = UNIBUS ADDRESS
58 007774 012760 000040 000010      MOV   #CLR,RMCS2(R0) ;CLEAR MASSBUS
59 010002 117760 171460 000010      MOVB  @TSTQUE,RMCS2(R0) ;SELECT DEVICE UNDER TEST
60 010010 016002 000026          MOV   RMDI(R0),R2   ;GET RMDI AND
61 010014 042702 177770          BIC   #177770,R2    ;SAVE DRIVE TYPE BITS
62 010020 022702 000007          CMP   #7,R2         ;IS IT AN RM05 ?
63 010024 001003          BNE   3$           ;NO, MUST BE AN RM02 OR RM03
64 010026 012737 011000 001334      MOV   #TA16!TA2,LSTRK ;YES--SET LAST TRACK - 18.
65
66          ;TYPE DRIVE NUMBER TO BE TESTED($UNIT)
67
68 010034 104401 001217 3$:      TYPE  , $CR LF      ;CR-LF
69 010040 104401 064342          TYPE  ,MSGDRV      ;TYPE 'DRIVE'
70 010044 013746 001234          MOV   $UNIT,-(SP)  ;;SAVE $UNIT FOR TYPEOUT
                          ;;TYPE DRIVE NUMBER
                          ;;GO TYPE--OCTAL ASCII
                          ;;TYPE 2 DIGIT(S)
                          ;;SUPPRESS LEADING ZEROS
                          ;;THESE TWO LOOPS ARE ADDED TO
                          ;;WAIT FOR TTY
      TYPOS
      .BYTE 2
      .BYTE 0
71 010054 005004          CLR   R4
72 010056 005304          DEC  R4
73 010060 001376          BNE  .-2
74 010062 005304          DEC  R4
75 010064 001376          BNE  .-2
  
```

*TEST 1 CONTROLLER ACCESS TEST

TST1:
010066
010066 000004
010070 000240
010072 012706 001100
010076 013700 001276
010102 013701 001466
010106 012737 000001 001226
2
3 010114 005001
4 010116 013746 000004
010122 013746 000006
5 010126 012737 010230 000004
6 010134 012737 000300 000006
7
8 010142 110160 000001
9 010146 010160 000002
10 010152 016002 000002
11 010156 010160 000004
12 010162 016002 000004
13 010166 016046 000010
14 010172 010160 000010
15 010176 016002 000010
16 010202 012660 000010
17 010206 010160 000022
18 010212 016002 000022
19 010216 012637 000006
010222 012637 000004
20 010226 000417
21
22 010230 022626
23 010232 012637 000006
010236 012637 000004
24 010242 104110
25 010244 005737 000042
26 010250 001002
27 010252 000137 005422
28 010256 005037 001300
29 010262 000137 031724
30 010266
31
32

SCOPE ;SCOPE CALL
NOP ;START OF TEST
MOV #STACK,SP ;INITIALIZE STACK POINTER
MOV \$BASE,R0 ;R0 = UNIBUS ADDRESS
MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
MOV #1,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX

CLR R1
MOV ERRVEC,-(SP) ;:PUSH ERRVEC ON STACK
MOV ERRVEC+2,-(SP) ;:PUSH ERRVEC+2 ON STACK
MOV #1\$,ERRVEC
MOV #PR6,ERRVEC+2

MOVB R1,RMCS1+1(R0) ;MOVE HI BYTE TO RMCS1
MOV R1,RMWC(R0) ;MOVE WORD COUNT REGISTER
MOV RMWC(R0),R2
MOV R1,RMBA(R0) ;MOVE BUS ADDRESS REGISTER
MOV RMBA(R0),R2
MOV RMCS2(R0),-(SP) ;:PUSH RMCS2(R0) ON STACK
MOV R1,RMCS2(R0) ;MOVE CONTROL STATUS REGISTER
MOV RMCS2(R0),R2
MOV (SP)+,RMCS2(R0) ;:POP STACK INTO RMCS2(R0)
MOV R1,RMDB(R0) ;MOVE DATA BUFFER
MOV RMDB(R0),R2
MOV (SP)+,ERRVEC+2 ;:POP STACK INTO ERRVEC+2
MOV (SP)+,ERRVEC ;:POP STACK INTO ERRVEC
BR 3\$;NO BUS TIMEOUT OCCURRED

1\$: CMP (SP)+,(SP)+ ;ADJUST STACK
MOV (SP)+,ERRVEC+2 ;:POP STACK INTO ERRVEC+2
MOV (SP)+,ERRVEC ;:POP STACK INTO ERRVEC
EMT 110
TST @#42 ;STAND ALONE MODE ?
BNE 2\$;NO!!
JMP START1 ;YES-GO GET \$BASE
2\$: CLR \$DEVM ;FUDGE NO DRIVES IN MAP
JMP \$EOP ;RETURN TO \$EOP
3\$:

*TEST 2 FORMAT ZEROS

TST2:
010266
010266 000004
010270 000240
010272 012706 001100
010276 013700 001276
010302 013701 001466
010306 012737 000002 001226
33
34
35 010314 012737 000000 001444
36 010322

SCOPE ;SCOPE CALL
NOP ;START OF TEST
MOV #STACK,SP ;INITIALIZE STACK POINTER
MOV \$BASE,R0 ;R0 = UNIBUS ADDRESS
MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
MOV #2,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX

;SETUP PARAMETERS FOR GENERATING DATA BUFFER
MOV #0,RMOFO ;18 BIT FORMAT
1\$:

```

37 010322 012737 000000 001446      MOV      #0.,RMDCO      :CYLINDER = 0
38 010330 012737 000000 001420      MOV      #0.,RMDAO      :TRACK = 0, SECTOR = 0
39 010336 012737 177376 001414      MOV      #-258.,RMWCO    :2 + 256 WORDS (2'S COMP)
40 010344 012737 101360 001416      MOV      #BUFONE,RMBAO   :DATA BUFFER ADDRESS
41 010352 012737 000062 001412      MOV      #WH,RMCS10     :WRITE HEADER AND DATA
42
43                                     ;VERIFY THAT SECTOR IS NOT BAD
      010360 004737 033152      JSR      PC,BADSCT      :CALL BAD SECTOR MODULE
      010364 000405                BR      2$              :GO TO 2$ IF NO ERROR
      010366 104401 063346      TYPE     ,SCTMSG       :TYPE BAD SECTOR MESSAGE
      010372 104000                EMT                    :ERROR # DEFINED BY BADSCT SUBROUTINE
      010374 000137 011130      JMP      20$           :GO TO 20$ IF ERROR
44 010400                                     2$:
45 010400 012737 065000 001174      MOV      #ZEROS,$TMP0   :USE ALL ZEROS DATA PATTERN
46 010406 012737 000001 001176      MOV      #1,$TMP1
47 010414 004737 035104      JSR      PC,GENBUF     :GO GENERATE DATA BUFFER
48 010420                                     3$:
49
50                                     ;PREPARE DEVICE FOR DATA TRANSFER
51 010420 004737 032176      JSR      PC,TSTPRP     :PREPARE DEVICE FOR TEST
      010424 154130      .WORD   154130        :TASK DESCRIPTOR AS FOLLOWS:
                                     :SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                     :CLEAR CONTROLLER & SELECT DEVICE
                                     :VERIFY CONTROLLER CLEAR OPERATION
                                     :PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                     :VERIFY PACK ACKNOWLEDGE
                                     :RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                     :VERIFY RECALIBRATION
      010426 000404                BR      4$              :GO TO 4$ IF NO ERROR
      010430 000240                NOP                    :RETURN HERE IF ERROR
      010432 104000                EMT                    :ERROR # DEFINED BY TSTPRP SUBROUTINE
      010434 000137 011130      JMP      20$           :GO TO 20$ IF ERROR
52 010440                                     4$:
53
54                                     ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
55 010440 012737 000005 001412      MOV      #SEEK.GO,RMCS10 :CHANGE COMMAND TO SEEK
56 010446 012702 001553                MOV      #PUTINX,R2     :WRITE REGISTER INDEX TABLE
57 010452 112722 000006                MOVVB   #RMDA,(R2)+
58 010456 112722 000034                MOVVB   #RMDC,(R2)+
59 010462 112722 000032                MOVVB   #RMOF,(R2)+
60 010466 112722 000000                MOVVB   #RMCS1,(R2)+
61 010472 112722 000200                MOVVB   #200,(R2)+
62
63 010476 004737 036240      JSR      PC,PUT        :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      010502 000404                BR      5$              :GO TO 5$ IF NO ERROR
      010504 000240                NOP                    :RETURN HERE IF ERROR
      010506 104000                EMT                    :ERROR # DEFINED BY PUT SUBROUTINE
      010510 000137 011130      JMP      20$           :GO TO 20$ IF ERROR
64 010514                                     5$:
65
66                                     ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
67 010514 004737 035704      JSR      PC,GETSTS     :SETUP FOR STATUS
68 010520 004737 036602      JSR      PC,TIMOUT    :WAIT FOR SEEK TO COMPLETE
69 010524                                     6$:
70
71                                     ;GO READ SEEK STATUS
72 010524 004737 035770      JSR      PC,GET        :GO READ REGISTER(S) WITH GET SUBROUTINE

```

010530 000404
 010532 000240
 010534 104000
 010536 000137 011130

```

BR      7$      :GO TO 7$ IF NO ERROR
NOP
EMT
JMP     20$     :RETURN HERE IF ERROR
                    :ERROR # DEFINED BY GET SUBROUTINE
                    :GO TO 20$ IF ERROR
    7$:
    
```

73 010542
 74
 75
 76 010542 004737 044104
 010546 000405
 010550 000240
 010552 104000
 010554 004736
 010556 000137 011130
 77 010562

```

:VERIFY THE RESULTS OF THE SEEK COMMAND
JSR    PC,SEKSTS :GO VERIFY RESULTS OF SEEK OPERATION
BR     8$        :GO TO 8$ IF NO ERROR
NOP
EMT
JSR    PC,@(SP)+ :RETURN HERE IF ERROR
                    :ERROR # DEFINED BY SEKSTS SUBROUTINE
                    :GO BACK FOR MORE ERROR CHECKS
                    :GO TO 20$ IF ERROR
    8$:
    
```

78
 79
 80 010562 012737 000063 001412
 81 010570 012702 001556
 82 010574 112722 000002
 83 010600 112722 000004
 84 010604 112722 000000
 85 010610 112722 000200
 86

```

:SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
MOV    #WH!GO,RMCS10 :WRITE HEADER AND DATA
MOV    #PUTINX+3,R2  :EXTEND REGISTER INDEX TABLE
MOVB  #RMWC,(R2)+
MOVB  #RMBA,(R2)+
MOVB  #RMCS1,(R2)+
MOVB  #200,(R2)+
    
```

87 010614 004737 036240
 010620 000404
 010622 000240
 010624 104000
 010626 000137 011130
 88 010632

```

JSR    PC,PUT     :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
BR     9$        :GO TO 9$ IF NO ERROR
NOP
EMT
JMP     20$     :RETURN HERE IF ERROR
                    :ERROR # DEFINED BY PUT SUBROUTINE
                    :GO TO 20$ IF ERROR
    9$:
    
```

89
 90
 91 010632 004737 036602
 92
 93 010636 004737 035770
 010642 000404
 010644 000240
 010646 104000
 010650 000137 011130
 94 010654

```

:WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
JSR    PC,TIMOUT  :WAIT FOR COMMAND TO COMPLETE

JSR    PC,GET     :GO READ REGISTER(S) WITH GET SUBROUTINE
BR     10$       :GO TO 10$ IF NO ERROR
NOP
EMT
JMP     20$     :RETURN HERE IF ERROR
                    :ERROR # DEFINED BY GET SUBROUTINE
                    :GO TO 20$ IF ERROR
    10$:
    
```

95
 96
 97 010654 004737 036766
 010660 000405
 010662 000240
 010664 104000
 010666 004736
 010670 000137 011130
 010674

```

:VERIFY RESULTS OF WRITE COMMAND
JSR    PC,PRIERR  :GO CHECK FOR PRIMARY ERRORS
BR     11$       :GO TO 11$ IF NO ERROR
NOP
EMT
JSR    PC,@(SP)+ :RETURN HERE IF ERROR
                    :ERROR # DEFINED BY PRIERR SUBROUTINE
                    :GO BACK FOR MORE ERROR CHECKS
                    :GO TO 20$ IF ERROR
    11$:
    
```

010674 004737 051502
 010700 000405
 010702 000240
 010704 104000
 010706 004736
 010710 000137 011130

```

JSR    PC,DTASTS :GO VERIFY RESULTS OF DATA TRANSFER
BR     12$       :GO TO 12$ IF NO ERROR
NOP
EMT
JSR    PC,@(SP)+ :RETURN HERE IF ERROR
                    :ERROR # DEFINED BY DTASTS SUBROUTINE
                    :GO BACK FOR MORE ERROR CHECKS
                    :GO TO 20$ IF ERROR
    12$:
    
```

100 010714
 101 010714 004737 037620
 010720 000405

```

JSR    PC,SECERR :GO CHECK FOR SECONDARY ERRORS
BR     13$       :GO TO 13$ IF NO ERROR
    
```

```

010722 000240      NOP      :RETURN HERE IF ERROR
010724 104000      EMT      :ERROR # DEFINED BY SECERR SUBROUTINE
010726 004736      JSR      PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
010730 000137 011130 JMP      20$      :GO TO 20$ IF ERROR
102 010734      13$:
103
104      :READ HEADER AND DATA FOR SECTOR JUST WRITTEN
105 010734 012737 000073 001412 MOV      #RH.GO,RMCS10 :READ HEADER & DATA COMMAND
106 010742 012737 102364 001416 MOV      #BUFTWO,RMBA0 :CHANGE BUS ADDRESS
107
108 010750 004737 036240 JSR      PC,PUT    :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    010754 000404      BR      14$      :GO TO 14$ IF NO ERROR
    010756 000240      NOP      :RETURN HERE IF ERROR
    010760 104000      EMT      :ERROR # DEFINED BY PUT SUBROUTINE
    010762 000137 011130 JMP      20$      :GO TO 20$ IF ERROR
109 010766      14$:
110
111      :WAIT FOR READ TO COMPLETE AND GET STATUS
112 010766 004737 036602 JSR      PC,TIMOUT :WAIT FOR READ TO COMPLETE
113
114 010772 004737 035770 JSR      PC,GET    :GO READ REGISTER(S) WITH GET SUBROUTINE
    010776 000404      BR      15$      :GO TO 15$ IF NO ERROR
    011000 000240      NOP      :RETURN HERE IF ERROR
    011002 104000      EMT      :ERROR # DEFINED BY GET SUBROUTINE
    011004 000137 011130 JMP      20$      :GO TO 20$ IF ERROR
115 011010      15$:
116
117      :VERIFY THE RESULTS OF READ OPERATION
118 011010 004737 036766 JSR      PC,PRIERR :GO CHECK FOR PRIMARY ERRORS
    011014 000405      BR      16$      :GO TO 16$ IF NO ERROR
    011016 000240      NOP      :RETURN HERE IF ERROR
    011020 104000      EMT      :ERROR # DEFINED BY PRIERR SUBROUTINE
    011022 004736      JSR      PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
    011024 000137 011130 JMP      20$      :GO TO 20$ IF ERROR
119 011030      16$:
120 011030 004737 051502 JSR      PC,DTASTS :GO VERIFY RESULTS OF DATA TRANSFER
    011034 000405      BR      17$      :GO TO 17$ IF NO ERROR
    011036 000240      NOP      :RETURN HERE IF ERROR
    011040 104000      EMT      :ERROR # DEFINED BY DTASTS SUBROUTINE
    011042 004736      JSR      PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
    011044 000137 011130 JMP      20$      :GO TO 20$ IF ERROR
121 011050      17$:
122 011050 004737 037620 JSR      PC,SECERR :GO CHECK FOR SECONDARY ERRORS
    011054 000405      BR      18$      :GO TO 18$ IF NO ERROR
    011056 000240      NOP      :RETURN HERE IF ERROR
    011060 104000      EMT      :ERROR # DEFINED BY SECERR SUBROUTINE
    011062 004736      JSR      PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
    011064 000137 011130 JMP      20$      :GO TO 20$ IF ERROR
123 011070      18$:
124
125      :VERIFY DATA
126 011070 004737 035342 JSR      PC,CMPBUF :GO COMPARE WRITE, READ DATA BUFFERS
    011074 101360      .WORD  BUFONE  :STARTING ADDRESS OF WRITE BUFFER
    011076 102364      .WORD  BUFTWO  :STARTING ADDRESS OF READ BUFFER
    011100 000402      BR      19$      :GO TO 19$ IF NO ERROR
    011102 000240      NOP      :RETURN HERE IF ERROR
    011104 104000      EMT      :ERROR # DEFINED BY CMPBUF SUBROUTINE

```

127 011106
 128 011106 032737 C10000 001444
 129 011114 001005
 130 011116 012737 010000 001444
 131 011124 000137 010322
 132 011130
 133
 134

19\$:
 BIT #FMT16,RMOFO ;TEST 16 BIT MODE YET ?
 BNE 20\$;YES
 MOV #FMT16,RMOFO ;SET 16 BIT MODE AND
 JMP 1\$;TEST AGAIN.
 20\$:

 ;*TEST 3 ZERO FILL TEST

 TST3:

011130
 011130 000004
 011132 000240
 011134 012706 001100
 011140 013700 001276
 011144 013701 001466
 011150 012737 000003 001226

SCOPE ;SCOPE CALL
 NOP ;START OF TEST
 MOV #STACK,SP ;INITIALIZE STACK POINTER
 MOV \$BASE,R0 ;R0 - UNIBUS ADDRESS
 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
 MOV #3,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX

135
 136
 137 011156 012737 000000 001446
 138 011164 012737 000000 001420
 139 011172 012737 010000 001444
 140 011200 012737 177376 001414
 141 011206 012737 101360 001416
 142 011214 012737 000062 001412
 143
 144

;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 MOV #0,RMDCO ;CYLINDER = 0
 MOV #0,RMDAO ;TRACK = 0, SECTOR = 0
 MOV #FMT16,RMOFO ;16 BIT FORMAT
 MOV #-258.,RMWCO ;2 + 256 WORDS (2'S COMP)
 MOV #BUFONE,RMBAD ;DATA BUFFER ADDRESS
 MOV #WH,RMCS10 ;WRITE HEADER AND DATA

011222 004737 033152
 011226 000405
 011230 104401 063346
 011234 104000
 011236 000137 011764

;VERIFY THAT SECTOR IS NOT BAD
 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
 BR 1\$;GO TO 1\$ IF NO ERROR
 TYPE .SCTMSG ;TYPE BAD SECTOR MESSAGE
 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
 JMP 18\$;GO TO 18\$ IF ERROR

145 011242
 146 011242 012737 065000 001174
 147 011250 012737 000001 001176
 148 011256 004737 035104
 149 011262
 150
 151

1\$:
 MOV #ZEROS,\$TMP0 ;USE ALL ZEROS DATA PATTERN
 MOV #1,\$TMP1
 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
 2\$:

152 011262 004737 032176
 011266 154130

;PREPARE DEVICE FOR DATA TRANSFER
 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
 ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
 ;CLEAR CONTROLLER & SELECT DEVICE
 ;VERIFY CONTROLLER CLEAR OPERATION
 ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
 ;VERIFY PACK ACKNOWLEDGE
 ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
 ;VERIFY RECALIBRATION

011270 000404
 011272 000240
 011274 104000
 011276 000137 011764
 153 011302
 154
 155

BR 3\$;GO TO 3\$ IF NO ERROR
 NOP ;RETURN HERE IF ERROR
 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 JMP 18\$;GO TO 18\$ IF ERROR
 3\$:

156 011302 012737 000005 001412
 157 011310 012702 001553

;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 MOV #SEEK.GO,R1,S10 ;CHANGE COMMAND TO SEEK
 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE

```

158 011314 112722 000006      MOVB  #RMDA,(R2)+
159 011320 112722 000034      MOVB  #RMDC,(R2)+
160 011324 112722 000032      MOVB  #RMOF,(R2)+
161 011330 112722 000000      MOVB  #RMCS1,(R2)+
162 011334 112722 000200      MOVB  #200,(R2)+
163
164 011340 004737 036240      JSR   PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      011344 000404      BR    4$             ;GO TO 4$ IF NO ERROR
      011346 000240      NOP                    ;RETURN HERE IF ERROR
      011350 104000      EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
      011352 000137 011764      JMP   18$           ;GO TO 18$ IF ERROR
165 011356      4$:
166
167      ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
168 011356 004737 035704      JSR   PC,GETSTS      ;SETUP FOR STATUS
169 011362 004737 036602      JSR   PC,TIMOUT      ;WAIT FOR SEEK TO COMPLETE
170 011366
171
172      ;GO READ SEEK STATUS
173 011366 004737 035770      JSR   PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      011372 000404      BR    6$             ;GO TO 6$ IF NO ERROR
      011374 000240      NOP                    ;RETURN HERE IF ERROR
      011376 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
      011400 000137 011764      JMP   18$           ;GO TO 18$ IF ERROR
174 011404
175
176      ;VERIFY THE RESULTS OF THE SEEK COMMAND
177 011404 004737 044104      JSR   PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
      011410 000405      BR    7$             ;GO TO 7$ IF NO ERROR
      011412 000240      NOP                    ;RETURN HERE IF ERROR
      011414 104000      EMT                    ;ERROR # DEFINED BY SEKSTS SUBROUTINE
      011416 004736      JSR   PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      011420 000137 011764      JMP   18$           ;GO TO 18$ IF ERROR
178 011424
179
180      ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
181 011424 012737 177776 001414      MOV   #-2,RMWC0      ;FORMAT PARTIAL SECTOR
182 011432 012737 000063 001412      MOV   #WH!GO,RMCS10  ;WRITE HEADER AND DATA
183 011440 012702 001556      MOV   #PUTINX+3,R2   ;EXTEND REGISTER INDEX TABLE
184 011444 112722 000002      MOVB  #RMWC,(R2)+
185 011450 112722 000004      MOVB  #RMSA,(R2)+
186 011454 112722 000000      MOVB  #RMCS1,(R2)+
187 011460 112722 000200      MOVB  #200,(R2)+
188
189 011464 004737 036240      JSR   PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      011470 000404      BR    8$             ;GO TO 8$ IF NO ERROR
      011472 000240      NOP                    ;RETURN HERE IF ERROR
      011474 104000      EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
      011476 000137 011764      JMP   18$           ;GO TO 18$ IF ERROR
190 011502      8$:
191
192      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
193 011502 004737 036602      JSR   PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
194
195 011506 004737 035770      JSR   PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      011512 000404      BR    9$             ;GO TO 9$ IF NO ERROR
      011514 000240      NOP                    ;RETURN HERE IF ERROR

```

```

011516 104000          EMT          :ERROR # DEFINED BY GET SUBROUTINE
011520 000137 011764  JMP          18$          :GO TO 18$ IF ERROR
196 011524          9$:
197
198
199 011524 004737 036766 :VERIFY RESULTS OF WRITE COMMAND
      JSR      PC,PRIERR      :GO CHECK FOR PRIMARY ERRORS
      BR       10$           :GO TO 10$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR      PC,@(SP)+      :GO BACK FOR MORE ERROR CHECKS
      JMP      18$           :GO TO 18$ IF ERROR
200 011544          10$:
201 011544 004737 051502 JSR      PC,DTASTS          :GO VERIFY RESULTS OF DATA TRANSFER
      BR       11$           :GO TO 11$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR      PC,@(SP)+      :GO BACK FOR MORE ERROR CHECKS
      JMP      18$           :GO TO 18$ IF ERROR
202 011564          11$:
203 011564 004737 037620 JSR      PC,SECERR          :GO CHECK FOR SECONDARY ERRORS
      BR       12$           :GO TO 12$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY SECERR SUBROUTINE
      JSR      PC,@(SP)+      :GO BACK FOR MORE ERROR CHECKS
      JMP      18$           :GO TO 18$ IF ERROR
204 011604          12$:
205
206 :READ HEADER AND DATA FOR SECTOR JUST WRITTEN
207 011604 012737 177376 001414 MOV      #-258.,RMWCO      :2 + 256 WORDS (2'S COMP)
208 011612 012737 000073 001412 MOV      #RH!GO,RMCS10    :READ HEADER & DATA COMMAND
209 011620 012737 102364 001416 MOV      #BUFTWO,RMBAO    :CHANGE BUS ADDRESS
210
211 011626 004737 036240 JSR      PC,PUT            :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      BR       13$           :GO TO 13$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY PUT SUBROUTINE
      JMP      18$           :GO TO 18$ IF ERROR
212 011644          13$:
213
214 :WAIT FOR READ TO COMPLETE AND GET STATUS
215 011644 004737 036602 JSR      PC,TIMOUT         :WAIT FOR READ TO COMPLETE
216
217 011650 004737 035770 JSR      PC,GET            :GO READ REGISTER(S) WITH GET SUBROUTINE
      BR       14$           :GO TO 14$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY GET SUBROUTINE
      JMP      18$           :GO TO 18$ IF ERROR
218 011666          14$:
219
220 :VERIFY THE RESULTS OF READ OPERATION
221 011666 004737 036766 JSR      PC,PRIERR          :GO CHECK FOR PRIMARY ERRORS
      BR       15$           :GO TO 15$ IF NO ERROR
      NOP      :RETURN HERE IF ERROR
      EMT      :ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR      PC,@(SP)+      :GO BACK FOR MORE ERROR CHECKS
      JMP      18$           :GO TO 18$ IF ERROR
222 011706          15$:

```

```

223 011706 004737 051502      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      011712 000405      BR     16$           ;GO TO 16$ IF NO ERROR
      011714 000240      NOP                    ;RETURN HERE IF ERROR
      011716 104000      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
      011720 004736      JSR    PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
      011722 000137 011764      JMP    18$           ;GO TO 18$ IF ERROR
224 011726                    16$:
225 011726 004737 037620      JSR    PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
      011732 000405      BR     17$           ;GO TO 17$ IF NO ERROR
      011734 000240      NOP                    ;RETURN HERE IF ERROR
      011736 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      011740 004736      JSR    PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
      011742 000137 011764      JMP    18$           ;GO TO 18$ IF ERROR
226 011746                    17$:
227
228
229 011746 004737 035342      ;VERIFY DATA
      011752 101360      JSR    PC,CMPBUF     ;GO COMPARE WRITE, READ DATA BUFFERS
      011754 102364      .WORD  BUFO1         ;STARTING ADDRESS OF WRITE BUFFER
      011756 000402      .WORD  BUFTWO        ;STARTING ADDRESS OF READ BUFFER
      011760 000240      BR     18$           ;GO TO 18$ IF NO ERROR
      011762 104000      NOP                    ;RETURN HERE IF ERROR
      011764                    EMT                    ;ERROR # DEFINED BY CMPBUF SUBROUTINE
230
231
232
      ;*****
      ;*TEST 4          FORMAT CHECK ZEROS
      ;*****
TST4:
      011764                    SCOPE                    ;SCOPE CALL
      011764 000004      NOP                    ;START OF TEST
      011766 000240      MOV     #STACK,SP          ;INITIALIZE STACK POINTER
      011770 012706 001100      MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
      011774 013700 001276      MOV     TSTQUE,R1         ;(R1) = DEVICE BEING TESTED
      012000 013701 001466      MOV     #4,$TESTN        ;SET TEST NUMBER IN APT MAIL BOX
      012004 012737 000004 001226
233
234
235 012012 012737 000000 001446 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
      012020 012737 000000 001420      MOV     #0,RMDCO        ;CYLINDER = 0
      012026 012737 010000 001444      MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
      012034 012737 177376 001414      MOV     #FMT16,RMOFO    ;16 BIT FORMAT
      012042 012737 101360 001416      MOV     #-258,RMWCO     ;2 + 256 WORDS (2'S COMP)
      012050 012737 000062 001412      MOV     #BUFO1,RMBAO    ;DATA BUFFER ADDRESS
      012056 004737 033152      MOV     #WH,RMCS10      ;WRITE HEADER AND DATA
      012062 000405
      012064 104401 063346
      012070 104000
      012072 000137 012554
243 012076                    ;VERIFY THAT SECTOR IS NOT BAD
244 012076 012737 065000 001174      JSR    PC,BADSCT     ;CALL BAD SECTOR MODULE
245 012104 012737 000001 001176      BR     1$           ;GO TO 1$ IF NO ERROR
246 012112 004737 035104      TYPE    ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
247 012116                    EMT                    ;ERROR # DEFINED BY BADSCT SUBROUTINE
248
249
250 012116 004737 032176      JMP    18$           ;GO TO 18$ IF ERROR
      1$:
      012116 012737 065000 001174      MOV     #ZEROS,$TMP0    ;USE ALL ZEROS DATA PATTERN
      012116 012737 000001 001176      MOV     #1,$TMP1
      012116 004737 035104      JSR    PC,GENBUF     ;GO GENERATE DATA BUFFER
251
252
253
254
255
256
257
258
259
260
      ;PREPARE DEVICE FOR DATA TRANSFER
      JSR    PC,TSTPRP  ;PREPARE DEVICE FOR TEST
  
```



```

012122 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 3$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY 'STPRP' SUBROUTINE
;GO TO 18$ IF ERROR

012124 000404 BR 3$
012126 000240 NOP
012130 104000 EMT
012132 000137 012554 JMP 18$
251 012136 3$:
252
253 ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
254 012136 012737 000005 001412 MOV #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
255 012144 012702 001553 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
256 012150 112722 000006 MOVB #RMDA,(R2)+
257 012154 112722 000034 MOVB #RMDC,(R2)+
258 012160 112722 000032 MOVB #RMOF,(R2)+
259 012164 112722 000000 MOVB #RMCS1,(R2)+
260 012170 112722 000200 MOVB #200,(R2)+
261
262 012174 004737 036240 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
012200 000404 BR 4$ ;GO TO 4$ IF NO ERROR
012202 000240 NOP ;RETURN HERE IF ERROR
012204 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
012206 000137 012554 JMP 18$ ;GO TO 18$ IF ERROR
263 012212 4$:
264
265 ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
266 012212 004737 035704 JSR PC,GETSTS ;SETUP FOR STATUS
267 012216 004737 036602 JSR PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE
268 012222 5$:
269
270 ;GO READ SEEK STATUS
271 012222 004737 035770 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
012226 000404 BR 6$ ;GO TO 6$ IF NO ERROR
012230 000240 NOP ;RETURN HERE IF ERROR
012232 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
012234 000137 012554 JMP 18$ ;GO TO 18$ IF ERROR
272 012240 6$:
273
274 ;VERIFY THE RESULTS OF THE SEEK COMMAND
275 012240 004737 044104 JSR PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
012244 000405 BR 7$ ;GO TO 7$ IF NO ERROR
012246 000240 NOP ;RETURN HERE IF ERROR
012250 104000 EMT ;ERROR # DEFINED BY SEKSTS SUBROUTINE
012252 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
012254 000137 012554 JMP 18$ ;GO TO 18$ IF ERROR
276 012260 7$:
277
278 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
279 012260 012737 000063 001412 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA
280 012266 012702 001556 MOV #PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
281 012272 112722 000002 MOVB #RMWC,(R2)+
282 012276 112722 000004 MOVB #RMBA,(R2)+
  
```

```

283 012302 112722 000000      MOVB  #RMCS1,(R2)+
284 012306 112722 C00200      MOVB  #200,(R2)+
285
286 012312 004737 036240      JSR   PC,PUT                ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      012316 000404          BR    8$                    ;GO TO 8$ IF NO ERROR
      012320 000240          NOP                   ;RETURN HERE IF ERROR
      012322 104000          EMT                   ;ERROR # DEFINED BY PUT SUBROUTINE
      012324 000137 012554      JMP   18$                   ;GO TO 18$ IF ERROR
287 012330      8$:
288
289      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
290 012330 004737 036602      JSR   PC,TIMOUT            ;WAIT FOR COMMAND TO COMPLETE
291
292 012334 004737 035770      JSR   PC,GET                ;GO READ REGISTER(S) WITH GET SUBROUTINE
      012340 000404          BR    9$                    ;GO TO 9$ IF NO ERROR
      012342 000240          NOP                   ;RETURN HERE IF ERROR
      012344 104000          EMT                   ;ERROR # DEFINED BY GET SUBROUTINE
      012346 000137 012554      JMP   18$                   ;GO TO 18$ IF ERROR
293 012352      9$:
294
295      ;VERIFY RESULTS OF WRITE COMMAND
296 012352 004737 036766      JSR   PC,PRIERR            ;GO CHECK FOR PRIMARY ERRORS
      012356 000405          BR    10$                   ;GO TO 10$ IF NO ERROR
      012360 000240          NOP                   ;RETURN HERE IF ERROR
      012362 104000          EMT                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
      012364 004736          JSR   PC,@(SP)+            ;GO BACK FOR MORE ERROR CHECKS
      012366 000137 012554      JMP   18$                   ;GO TO 18$ IF ERROR
297 012372      10$:
298 012372 004737 051502      JSR   PC,DTASTS            ;GO VERIFY RESULTS OF DATA TRANSFER
      012376 000405          BR    11$                    ;GO TO 11$ IF NO ERROR
      012400 000240          NOP                   ;RETURN HERE IF ERROR
      012402 104000          EMT                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
      012404 004736          JSR   PC,@(SP)+            ;GO BACK FOR MORE ERROR CHECKS
      012406 000137 012554      JMP   18$                   ;GO TO 18$ IF ERROR
299 012412      11$:
300 012412 004737 037620      JSR   PC,SECERR            ;GO CHECK FOR SECONDARY ERRORS
      012416 000405          BR    12$                    ;GO TO 12$ IF NO ERROR
      012420 000240          NOP                   ;RETURN HERE IF ERROR
      012422 104000          EMT                   ;ERROR # DEFINED BY SECERR SUBROUTINE
      012424 004736          JSR   PC,@(SP)+            ;GO BACK FOR MORE ERROR CHECKS
      012426 000137 012554      JMP   18$                   ;GO TO 18$ IF ERROR
301 012432      12$:
302
303      ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
304 012432 012737 000053 001412  MOV   #WCH!GO,RMCS10      ;WRITE CHECK COMMAND
305
306 012440 004737 036240      JSR   PC,PUT                ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      012444 000404          BR    13$                    ;GO TO 13$ IF NO ERROR
      012446 000240          NOP                   ;RETURN HERE IF ERROR
      012450 104000          EMT                   ;ERROR # DEFINED BY PUT SUBROUTINE
      012452 000137 012554      JMP   18$                   ;GO TO 18$ IF ERROR
307 012456      13$:
308
309      ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
310 012456 004737 036602      JSR   PC,TIMOUT            ;WAIT FOR READ TO COMPLETE
311
312 012462 004737 035770      JSR   PC,GET                ;GO READ REGISTER(S) WITH GET SUBROUTINE
  
```

```
012466 000404 BR 14$ :GO TO 14$ IF NO ERROR
012470 000240 NOP :RETURN HERE IF ERROR
012472 104000 EMT :ERROR # DEFINED BY GET SUBROUTINE
012474 000137 012554 JMP 18$ :GO TO 18$ IF ERROR
313 012500 14$:
314
315 :VERIFY THE RESULTS OF WRITE CHECK OPERATION
316 012500 004737 036766 JSR PC,PRIERR :GO CHECK FOR PRIMARY ERRORS
012504 000405 BR 15$ :GO TO 15$ IF NO ERROR
012506 000240 NOP :RETURN HERE IF ERROR
012510 104000 EMT :ERROR # DEFINED BY PRIERR SUBROUTINE
012512 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
012514 000137 012554 JMP 18$ :GO TO 18$ IF ERROR
317 012520 15$:
318 012520 004737 051502 JSR PC,DTASTS :GO VERIFY RESULTS OF DA A TRANSFER
012524 000405 BR 16$ :GO TO 16$ IF NO ERROR
012526 000240 NOP :RETURN HERE IF ERROR
012530 104000 EMT :ERROR # DEFINED BY DTASTS SUBROUTINE
012532 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
012534 000137 012554 JMP 18$ :GO TO 18$ IF ERROR
319 012540 16$:
320 012540 004737 037620 JSR PC,SECERR :GO CHECK FOR SECONDARY ERRORS
012544 000403 BR 17$ :GO TO 17$ IF NO ERROR
012546 000240 NOP :RETURN HERE IF ERROR
012550 104000 EMT :ERROR # DEFINED BY SECERR SUBROUTINE
012552 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
321 012554 17$:
322
323 012554 18$:
324
325
:*****
:*TEST 5 FORMAT CHECK ZEROS W/ WCE ERROR
:*****
TST5:
012554 SCOPE :SCOPE CALL
012554 000004 NOP :START OF TEST
012556 000240 MOV #STACK,SP :INITIALIZE STACK POINTER
012560 012706 001100 MOV $BASE,R0 :R0 = UNIBUS ADDRESS
012564 013700 001276 MOV TSTQUE,R1 : (R1) = DEVICE BEING TESTED
012570 013701 001466 MOV #5,$TESTN :SET TEST NUMBER IN APT MAIL BOX
012574 012737 000005 001226
326
327 :SETUP PARAMETERS FOR GENERATING DATA BUFFER
328 012602 012737 000000 001446 MOV #0,RMDCO :CYLINDER = 0
329 012610 012737 000000 001420 MOV #0,RMDAO :TRACK = 0, SECTOR = 0
330 012616 012737 010000 001444 MOV #FMT16,RMOFO :16 BIT FORMAT
331 012624 012737 177376 001414 MOV #-258.,RMWCO :2 + 256 WORDS (2'S COMP)
332 012632 012737 101360 001416 MOV #BUFONE,RMBAO :DATA BUFFER ADDRESS
333 012640 012737 000062 001412 MOV #WH,RMCS10 :WRITE HEADER AND DATA
334
335 :VERIFY THAT SECTOR IS NOT BAD
012646 004737 033152 JSR PC,BADSCT :CALL BAD SECTOR MODULE
012652 000405 BR 1$ :GO TO 1$ IF NO ERROR
012654 104401 063346 TYPE ,SCTMSG :TYPE BAD SECTOR MESSAGE
012660 104000 EMT :ERROR # DEFINED BY BADSCT SUBROUTINE
012662 000137 013514 JMP 20$ :GO TO 20$ IF ERROR
336 012666 1$:
337 012666 012737 065000 001174 MOV #ZEROS,$TMP0 :USE ALL ONES DATA PATTERN
```

```

338 012674 012737 000001 001176      MOV    #1,$TMP1
339 012702 004737 C35104      JSR    PC,GENBUF      ;GO GENERATE DATA BUFFER
340 012705                                2$:
341
342                                ;PREPARE DEVICE FOR DATA TRANSFER
343 012706 004737 032176      JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
      012712 154130      .WORD 154130          ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
      ;VERIFY PACK ACKNOWLEDGE
      ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      ;VERIFY RECALIBRATION
      ;GO TO 3$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY TSTPRP SUBROUTINE
      ;GO TO 20$ IF ERROR
      012714 000404      BR     3$
      012716 000240      NOP
      012720 104000      EMT
      012722 000137 013514  JMP    20$
344 012726                                3$:
345
346                                ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
347 012726 012737 000005 001412  MOV    #SEEK:GO,RMCS10 ;CHANGE COMMAND TO SEEK
348 012734 012702 001553      MOV    #PUTINX,R2     ;WRITE REGISTER INDEX TABLE
349 012740 112722 000006      MOVB  #RMDA,(R2)+
350 012744 112722 000034      MOVB  #RMDC,(R2)+
351 012750 112722 000032      MOVB  #RMOF,(R2)+
352 012754 112722 000000      MOVB  #RMCS1,(R2)+
353 012760 112722 000200      MOVB  #200,(R2)+
354
355 012764 004737 036240      JSR    PC,PUT         ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      012770 000404      BR     4$
      012772 000240      NOP
      012774 104000      EMT
      012776 000137 013514  JMP    20$
      ;GO TO 4$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY PUT SUBROUTINE
      ;GO TO 20$ IF ERROR
356 013002                                4$:
357
358                                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
359 013002 004737 035704      JSR    PC,GETSTS     ;SETUP FOR STATUS
360 013006 004737 036602      JSR    PC,TIMOUT     ;WAIT FOR SEEK TO COMPLETE
361 013012                                5$:
362
363                                ;GO READ SEEK STATUS
364 013012 004737 035770      JSR    PC,GET        ;GO READ REGISTER(S) WITH GET SUBROUTINE
      013016 000404      BR     6$
      013020 000240      NOP
      013022 104000      EMT
      013024 000137 013514  JMP    20$
      ;GO TO 6$ IF NO ERROR
      ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY GET SUBROUTINE
      ;GO TO 20$ IF ERROR
365 013030                                6$:
366
367                                ;VERIFY THE RESULTS OF THE SEEK COMMAND
368 013030 004737 044104      JSR    PC,SEKSTS     ;GO VERIFY RESULTS OF SEEK OPERATION
      013034 000405      BR     7$
      013036 000240      NOP
      013040 104000      EMT
      013042 004736      JSR    PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
      013044 000137 013514  JMP    20$
      ;GO TO 20$ IF ERROR
369 013050                                7$:
  
```

```

370
371 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
372 013050 012737 000063 001412 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA
373 013056 012702 001556 MOV #PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
374 013062 112722 000002 MOVB #RMWC,(R2)+
375 013066 112722 000004 MOVB #RMBA,(R2)+
376 013072 112722 000000 MOVB #RMCS1,(R2)+
377 013076 112722 000200 MOVB #200,(R2)+
378
379 013102 004737 036240 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    013106 000404 BR 8$ ;GO TO 8$ IF NO ERROR
    013110 000240 NOP ;RETURN HERE IF ERROR
    013112 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
    013114 000137 013514 JMP 20$ ;GO TO 20$ IF ERROR
380 013120 8$:
381
382 ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
383 013120 004737 036602 JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
384
385 C13124 004737 035770 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
    013130 000404 BR 9$ ;GO TO 9$ IF NO ERROR
    013132 000240 NOP ;RETURN HERE IF ERROR
    013134 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
    013136 000137 013514 JMP 20$ ;GO TO 20$ IF ERROR
386 013142 9$:
387
388 ;VERIFY RESULTS OF WRITE COMMAND
389 013142 004737 036766 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
    013146 000405 BR 10$ ;GO TO 10$ IF NO ERROR
    013150 000240 NOP ;RETURN HERE IF ERROR
    013152 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
    013154 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    013156 000137 013514 JMP 20$ ;GO TO 20$ IF ERROR
390 013162 10$:
391 013162 004737 051502 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
    013166 000405 BR 11$ ;GO TO 11$ IF NO ERROR
    013170 000240 NOP ;RETURN HERE IF ERROR
    013172 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
    013174 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    013176 000137 013514 JMP 20$ ;GO TO 20$ IF ERROR
392 013202 11$:
393 013202 004737 037620 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
    013206 000405 BR 12$ ;GO TO 12$ IF NO ERROR
    013210 000240 NOP ;RETURN HERE IF ERROR
    013212 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
    013214 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    013216 000137 013514 JMP 20$ ;GO TO 20$ IF ERROR
394 013222 12$:
395
396 ;ALTER DATA BUFFER
397 013222 005137 102362 COM BUFTWO-2 ;COMPLEMENT LAST DATA WORD
398
399 ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
400 013226 012737 000053 001412 MOV #WCH.GO,RMCS10 ;WRITE CHECK COMMAND
401
402 013234 004737 036240 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    013240 000404 BR 13$ ;GO TO 13$ IF NO ERROR
  
```

```

013242 000240      NOP      ;RETURN HERE IF ERROR
013244 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
013246 000137 013514  JMP      20$      ;GO TO 20$ IF ERROR
403 013252      13$:
404
405      ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
406 013252 004737 036602  JSR      PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
407
408 013256 004737 035770  JSR      PC,GET    ;GO READ REGISTER(S) WITH GET SUBROUTINE
013262 000404      BR      14$      ;GO TO 14$ IF NO ERROR
013264 000240      NOP      ;RETURN HERE IF ERROR
013266 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
013270 000137 013514  JMP      20$      ;GO TO 20$ IF ERROR
409 013274      14$:
410
411      ;CHECK FOR PRIMARY ERRORS
412 013274 004737 036766  JSR      PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
013300 000405      BR      15$      ;GO TO 15$ IF NO ERROR
013302 000240      NOP      ;RETURN HERE IF ERROR
013304 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
013306 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
013310 000137 013514  JMP      20$      ;GO TO 20$ IF ERROR
413 013314      15$:
414
415      ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
416 013314 032737 040000 001346  BIT      #WCE,RMCS2I ;IS WRITE CHECK ERROR SET??
417 013322 001023      BNE      17$      ;YES!!
418
419 013324 004737 051502  JSR      PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
013330 000405      BR      16$      ;GO TO 16$ IF NO ERROR
013332 000240      NOP      ;RETURN HERE IF ERROR
013334 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
013336 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
013340 000137 013514  JMP      20$      ;GO TO 20$ IF ERROR
420
421 013344 013737 001346 001140 16$:  MOV      RMCS2I,$GDDAT ;LOAD EXPECTED STATUS
422 013352 052737 040000 001140  BIS      #WCE,$GDDAT
423 013360 013737 001346 001142  MOV      RMCS2I,$BDDAT ;LOAD RECEIVED STATUS
424 013366 104337      EMT      337
425 013370 000451      BR      20$
426 013372      17$:
427
428      ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
429 013372 012737 102362 001134  MOV      #BUFTWO-2,$GDADR ;LOAD EXPECTED ADDRESS
430 013400 013737 001342 001136  MOV      RMBAI,$BDADR   ;LOAD RECEIVED ADDRESS
431 013406 162737 000002 001136  SUB      #2,$BDADR      ;DECREMENT RECEIVED ADDRESS
432
433      ;GET WCE DATA AND VERIFY IT IS OK
434 013414 112737 000022 001524  MOVSB   #RMDB,GETINX   ;SETUP FOR READING RMDB
435 013422 112737 000200 001525  MOVSB   #200,GETINX+1
436
437 013430 004737 035770  JSR      PC,GET    ;GO READ REGISTER(S) WITH GET SUBROUTINE
013434 000404      BR      18$      ;GO TO 18$ IF NO ERROR
013436 000240      NOP      ;RETURN HERE IF ERROR
013440 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
013442 000137 013514  JMP      20$      ;GO TO 20$ IF ERROR
438 013446 013737 001360 001142 18$:  MOV      RMDBI,$BDDAT ;LOAD RECEIVED DATA WORD
    
```

```

439 013454 013737 102362 001140      MOV      BUF-TWO-2,$GDDAT ;LOAD EXPECTED DATA WORD
440 013462 005137 001140      COM      $GDDAT
441 013466 023737 001134 001136      CMP      $GDADR,$BDADR ;IS ADDRESS OK??
442 013474 001402      BEQ      19$ ;YES..
443 013476 104340      EMT      340
444 013500 000405      BR       20$
445 013502 023737 001140 001142 19$:      CMP      $GDDAT,$BDDAT ;IS DATA WORD OK??
446 013510 001401      BEQ      20$ ;YES..
447 013512 104341      EMT      341
448 013514      20$:
449
450
;*****
;*TEST 6          FORMAT ONES
;*****
TST6:
013514
013514 000004      SCOPE
013516 000240      NOP ;SCOPE CALL
013520 012706 001100      MOV      #STACK,SP ;START OF TEST
013524 013700 001276      MOV      $BASE,R0 ;INITIALIZE STACK POINTER
013530 013701 001466      MOV      TSTQUE,R1 ;R0 UNIBUS ADDRESS
013534 012737 000006 001226      MOV      #6,$TESTN ;(R1) = DEVICE BEING TESTED
;:SET TEST NUMBER IN APT MAIL BOX

;SETUP PARAMETERS FOR GENERATING DATA BUFFER
451
452
453 013542 012737 000000 001446      MOV      #0,RMDCO ;CYLINDER - 0
454 013550 012737 000000 001420      MOV      #0,RMDAO ;TRACK = 0, SECTOR - 0
455 013556 012737 010000 001444      MOV      #FMT16,RMFCO ;16 BIT FORMAT
456 013564 012737 177376 001414      MOV      #-258,RMWCO ;2 + 256 WORDS (2'S COMP)
457 013572 012737 101360 001416      MOV      #BUFONE,RMBAO ;DATA BUFFER ADDRESS
458 013600 012737 000062 001412      MOV      #WH,RMC$10 ;WRITE HEADER AND DATA
459
460
;VERIFY THAT SECTOR IS NOT BAD
013606 004737 033152      JSR      PC,BADSCT ;CALL BAD SECTOR MODULE
013612 000405      BR       1$ ;GO TO 1$ IF NO ERROR
013614 104401 063346      TYPE     ,SCTMSG ;TYPE BAD SECTOR MESSAGE
013620 104000      EMT
013622 000137 014334      JMP      18$ ;ERROR # DEFINED BY BADSCT SUBROUTINE
;GO TO 18$ IF ERROR
461 013626      1$:
462 013626 012737 064736 001174      MOV      #ONES,$TMP0 ;USE ALL ONES DATA PATTERN
463 013634 012737 000001 001176      MOV      #1,$TMP1
464 013642 004737 035104      JSR      PC,GENBUF ;GO GENERATE DATA BUFFER
465 013646      2$:
466
467
;PREPARE DEVICE FOR DATA TRANSFER
468 013646 004737 032176      JSR      PC,TSTPRP ;PREPARE DEVICE FOR TEST
013652 154130      .WORD   154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
013654 000404      BR       3$ ;GO TO 3$ IF NO ERROR
013656 000240      NOP ;RETURN HERE IF ERROR
013660 104000      EMT
013662 000137 014334      JMP      18$ ;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 18$ IF ERROR
469 013666      3$:

```

```

470
471
472 013666 012737 000005 0014*2 ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
473 013674 012702 001553          MOV #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
474 013700 112722 000006          MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
475 013704 112722 000034          MOV #RMDA,(R2)+
476 013710 112722 000032          MOV #RMDC,(R2)+
477 013714 112722 000000          MOV #RMOF,(R2)+
478 013720 112722 000200          MOV #RMCS1,(R2)+
479
480 013724 004737 036240          JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      013730 000404          BR 4$ ;GO TO 4$ IF NO ERROR
      013732 000240          NOP ;RETURN HERE IF ERROR
      013734 104000          EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      013736 000137 014334          JMP 18$ ;GO TO 18$ IF ERROR
481 013742
482
483 ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
484 013742 004737 035704          JSR PC,GETSTS ;SETUP FOR STATUS
485 013746 004737 036602          JSR PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE
486 013752
487
488 ;GO READ SEEK STATUS
489 013752 004737 035770          JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      013756 000404          BR 6$ ;GO TO 6$ IF NO ERROR
      013760 000240          NOP ;RETURN HERE IF ERROR
      013762 104000          EMT ;ERROR # DEFINED BY GET SUBROUTINE
      013764 000137 014334          JMP 18$ ;GO TO 18$ IF ERROR
490 013770
491
492 ;VERIFY THE RESULTS OF THE SEEK COMMAND
493 013770 004737 044104          JSR PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
      013774 000405          BR 7$ ;GO TO 7$ IF NO ERROR
      013776 000240          NOP ;RETURN HERE IF ERROR
      014000 104000          EMT ;ERROR # DEFINED BY SEKSTS SUBROUTINE
      014002 004736          JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      014004 000137 014334          JMP 18$ ;GO TO 18$ IF ERROR
494 014010
495
496 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
497 014010 012737 000063 001412          MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA
498 014016 012702 001556          MOV #PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
499 014022 112722 000002          MOV #RMWC,(R2)+
500 014026 112722 000004          MOV #RMBB,(R2)+
501 014032 112722 000000          MOV #RMCS1,(R2)+
502 014036 112722 000200          MOV #200,(R2)+
503
504 014042 004737 036240          JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      014046 000404          BR 8$ ;GO TO 8$ IF NO ERROR
      014050 000240          NOP ;RETURN HERE IF ERROR
      014052 104000          EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      014054 000137 014334          JMP 18$ ;GO TO 18$ IF ERROR
505 014060
506
507 ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
508 014060 004737 036602          JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
509

```



```

510 014064 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      014070 000404      BR      9$        ;GO TO 9$ IF NO ERROR
      014072 000240      NOP      ;RETURN HERE IF ERROR
      014074 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      014076 000137 014334  JMP      18$      ;GO TO 18$ IF ERROR
511 014102      9$:
512
513      ;VERIFY RESULTS OF WRITE COMMAND
514 014102 004737 036766      JSR    PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      014106 000405      BR      10$       ;GO TO 10$ IF NO ERROR
      014110 000240      NOP      ;RETURN HERE IF ERROR
      014112 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      014114 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      014116 000137 014334  JMP      18$       ;GO TO 18$ IF ERROR
515 014122      10$:
516 014122 004737 051502      JSR    PC,DTASTS   ;GO VERIFY RESULTS OF DATA TRANSFER
      014126 000405      BR      11$       ;GO TO 11$ IF NO ERROR
      014130 000240      NOP      ;RETURN HERE IF ERROR
      014132 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      014134 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      014136 000137 014334  JMP      18$       ;GO TO 18$ IF ERROR
517 014142      11$:
518 014142 004737 037620      JSR    PC,SELERR   ;GO CHECK FOR SECONDARY ERRORS
      014146 000405      BR      12$       ;GO TO 12$ IF NO ERROR
      014150 000240      NOP      ;RETURN HERE IF ERROR
      014152 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      014154 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      014156 000137 014334  JMP      18$       ;GO TO 18$ IF ERROR
519 014162      12$:
520
521      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
522 014162 012737 000073 001412  MOV     #RH!GO,RMCS10 ;READ HEADER & DATA COMMAND
523 014170 012737 102364 001416  MOV     #BUFTWO,RMBA0 ;CHANGE BUS ADDRESS
524
525 014176 004737 036240      JSR    PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      014202 000404      BR      13$       ;GO TO 13$ IF NO ERROR
      014204 000240      NOP      ;RETURN HERE IF ERROR
      014206 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      014210 000137 014334  JMP      18$       ;GO TO 18$ IF ERROR
526 014214      13$:
527
528      ;WAIT FOR READ TO COMPLETE AND GET STATUS
529 014214 004737 036602      JSR    PC,TIMOUT   ;WAIT FOR READ TO COMPLETE
530
531 014220 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      014224 000404      BR      14$       ;GO TO 14$ IF NO ERROR
      014226 000240      NOP      ;RETURN HERE IF ERROR
      014230 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      014232 000137 014334  JMP      18$       ;GO TO 18$ IF ERROR
532 014236      14$:
533
534      ;VERIFY THE RESULTS OF READ OPERATION
535 014236 004737 036766      JSR    PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      014242 000405      BR      15$       ;GO TO 15$ IF NO ERROR
      014244 000240      NOP      ;RETURN HERE IF ERROR
      014246 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      014250 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS

```

```

536 014252 000137 014334          JMP      18$          :GO TO 18$ IF ERROR
537 014256 004737 051502          JSR      PC,DTASTS   :GO VERIFY RESULTS OF DATA TRANSFER
    014262 000405          BR       16$          :GO TO 16$ IF NO ERROR
    014264 000240          NOP                     :RETURN HERE IF ERROR
    014266 104000          EMT                     :ERROR # DEFINED BY DTASTS SUBROUTINE
    014270 004736          JSR      PC,@(SP)+   :GO BACK FOR MORE ERROR CHECKS
    014272 000137 014334          JMP      18$          :GO TO 18$ IF ERROR
538 014276 004737 037620          JSR      PC,SECERR   :GO CHECK FOR SECONDARY ERRORS
539 014302 000405          BR       17$          :GO TO 17$ IF NO ERROR
    014304 000240          NOP                     :RETURN HERE IF ERROR
    014306 104000          EMT                     :ERROR # DEFINED BY SECERR SUBROUTINE
    014310 004736          JSR      PC,@(SP)+   :GO BACK FOR MORE ERROR CHECKS
    014312 000137 014334          JMP      18$          :GO TO 18$ IF ERROR
540 014316 004737 035342          JSR      PC,CMPBUF   :GO COMPARE WRITE, READ DATA BUFFERS
541 014322 101360          .WORD   BUFONE        :STARTING ADDRESS OF WRITE BUFFER
542 014324 102364          .WORD   BUFTWO        :STARTING ADDRESS OF READ BUFFER
543 014326 000402          BR       18$          :GO TO 18$ IF NO ERROR
    014330 000240          NOP                     :RETURN HERE IF ERROR
    014332 104000          EMT                     :ERROR # DEFINED BY CMPBUF SUBROUTINE
544 014334
545
546
;*****
;*TEST 7          FORMAT CHECK ONES
;*****
TST7:
    014334 000004          NOP                     :SCOPE CALL
    014336 000240          MOV      #STACK,SP    :START OF TEST
    014340 012706 001100      MOV      $BASE,R0      :INITIALIZE STACK POINTER
    014344 013700 001276      MOV      TSTQUE,R1     :R0 = UNIBUS ADDRESS
    014350 013701 001466      MOV      #7,$TESTN     :(R1) = DEVICE BEING TESTED
    014354 012737 000007 001226  MOV      #7,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

;SETUP PARAMETERS FOR GENERATING DATA BUFFER
549 014362 012737 000000 001446      MOV      #0,RMDCO      :CYLINDER = 0
550 014370 012737 000000 001420      MOV      #0,RMDAO      :TRACK = 0, SECTOR = 0
551 014376 012737 010000 001444      MOV      #FMT16,RMOFO  :16 BIT FORMAT
552 014404 012737 177376 001414      MOV      #-258,RMWCO   :2 + 256 WORDS (2'S COMP)
553 014412 012737 101360 001416      MOV      #BUFONE,RMBAO :DATA BUFFER ADDRESS
554 014420 012737 000062 001412      MOV      #WH,RMCS10    :WRITE HEADER AND DATA
555
556
;VERIFY THAT SECTOR IS NOT BAD
    014426 004737 033152          JSR      PC,BADSCT    :CALL BAD SECTOR MODULE
    014432 000405          BR       1$          :GO TO 1$ IF NO ERROR
    014434 104401 063346          TYPE    ,SCTMSG      :TYPE BAD SECTOR MESSAGE
    014440 104000          EMT                     :ERROR # DEFINED BY BADSCT SUBROUTINE
    014442 000137 015124          JMP      18$          :GO TO 18$ IF ERROR
557 014446 012737 064736 001174          MOV      #ONES,$TMP0   :USE ALL ONES DATA PATTERN
558 014446 012737 000001 001176          MOV      #1,$TMP1
559 014454 012737 000001 001176          JSR      PC,GENBUF    :GO GENERATE DATA BUFFER
560 014462 004737 035104
561 014466
562

```

```

563                                     ;PREPARE DEVICE FOR DATA TRANSFER
564 014466 004737 032176                JSR   PC,TSTPRP                ;PREPARE DEVICE FOR TEST
    014472 154130                        .WORD 154130                    ;TASK DESCRIPTOR AS FOLLOWS:
                                     ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                     ;CLEAR CONTROLLER & SELECT DEVICE
                                     ;VERIFY CONTROLLER CLEAR OPERATION
                                     ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                     ;VERIFY PACK ACKNOWLEDGE
                                     ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                     ;VERIFY RECALIBRATION
                                     ;GO TO 3$ IF NO ERROR
                                     ;RETURN HERE IF ERROR
                                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                     ;GO TO 18$ IF ERROR

    014474 000404                        BR    3$
    014476 000240                        NOP
    014500 104000                        EMT
    014502 000137 015124                JMP   18$
565 014506                               3$:
566
567                                     ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
568 014506 012737 000005 001412        MOV   #SEEK.GO,RMCS10        ;CHANGE COMMAND TO SEEK
569 014514 012702 001553                MOV   #PUTINX,R2            ;WRITE REGISTER INDEX TABLE
570 014520 112722 000006                MOVB  #RMDA,(R2)+
571 014524 112722 000034                MOVB  #RMDC,(R2)+
572 014530 112722 000032                MOVB  #RMOF,(R2)+
573 014534 112722 000000                MOVB  #RMCS1,(R2)+
574 014540 112722 000200                MOVB  #200,(R2)+
575
576 014544 004737 036240                JSR   PC,PUT                ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    014550 000404                        BR    4$
    014552 000240                        NOP
    014554 104000                        EMT
    014556 000137 015124                JMP   18$
577 014562                               4$:
578
579                                     ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
580 014562 004737 035704                JSR   PC,GETSTS            ;SETUP FOR STATUS
581 014566 004737 036602                JSR   PC,TIMOUT           ;WAIT FOR SEEK TO COMPLETE
582 014572
583
584                                     ;GO READ SEEK STATUS
585 014572 004737 035770                JSR   PC,GET                ;GO READ REGISTER(S) WITH GET SUBROUTINE
    014576 000404                        BR    6$
    014600 000240                        NOP
    014602 104000                        EMT
    014604 000137 015124                JMP   18$
586 014610                               6$:
587
588                                     ;VERIFY THE RESULTS OF THE SEEK COMMAND
589 014610 004737 044104                JSR   PC,SEKSTS           ;GO VERIFY RESULTS OF SEEK OPERATION
    014614 000405                        BR    7$
    014616 000240                        NOP
    014620 104000                        EMT
    014622 004736                        JSR   PC,@(SP)+           ;GO BACK FOR MORE ERROR CHECKS
    014624 000137 015124                JMP   18$
590 014630                               7$:
591
592                                     ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
593 014630 012737 000063 001412        MOV   #WH.GO,RMCS10        ;WRITE HEADER AND DATA
594 014636 012702 001556                MOV   #PUTINX+3,R2        ;EXTEND REGISTER INDEX TABLE
  
```

```

595 014642 112722 000002      MOVB  #RMWC,(R2)+
596 014646 112722 000004      MOVB  #RMBB,(R2)+
597 014652 112722 000000      MOVB  #RMCS1,(R2)+
598 014656 112722 000200      MOVB  #200,(R2)+
599
600 014662 004737 036240      JSR   PC,PUT                ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    014666 000404                BR    8$                    ;GO TO 8$ IF NO ERROR
    014670 000240                NOP                    ;RETURN HERE IF ERROR
    014672 104000                EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
    014674 000137 015124        JMP   18$                  ;GO TO 18$ IF ERROR
601 014700      8$:
602
603      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
604 014700 004737 036602      JSR   PC,TIMOUT            ;WAIT FOR COMMAND TO COMPLETE
605
606 014704 004737 035770      JSR   PC,GET                ;GO READ REGISTER(S) WITH GET SUBROUTINE
    014710 000404                BR    9$                    ;GO TO 9$ IF NO ERROR
    014712 000240                NOP                    ;RETURN HERE IF ERROR
    014714 104000                EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
    014716 000137 015124        JMP   18$                  ;GO TO 18$ IF ERROR
607 014722      9$:
608
609      ;VERIFY RESULTS OF WRITE COMMAND
610 014722 004737 036766      JSR   PC,PRIERR            ;GO CHECK FOR PRIMARY ERRORS
    014726 000405                BR    10$                   ;GO TO 10$ IF NO ERROR
    014730 000240                NOP                    ;RETURN HERE IF ERROR
    014732 104000                EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
    014734 004736                JSR   PC,@(SP)+            ;GO BACK FOR MORE ERROR CHECKS
    014736 000137 015124        JMP   18$                  ;GO TO 18$ IF ERROR
611 014742      10$:
612 014742 004737 051502      JSR   PC,DTASTS            ;GO VERIFY RESULTS OF DATA TRANSFER
    014746 000405                BR    11$                   ;GO TO 11$ IF NO ERROR
    014750 000240                NOP                    ;RETURN HERE IF ERROR
    014752 104000                EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
    014754 004736                JSR   PC,@(SP)+            ;GO BACK FOR MORE ERROR CHECKS
    014756 000137 015124        JMP   18$                  ;GO TO 18$ IF ERROR
613 014762      11$:
614 014762 004737 037620      JSR   PC,SECERR            ;GO CHECK FOR SECONDARY ERRORS
    014766 000405                BR    12$                   ;GO TO 12$ IF NO ERROR
    014770 000240                NOP                    ;RETURN HERE IF ERROR
    014772 104000                EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
    014774 004736                JSR   PC,@(SP)+            ;GO BACK FOR MORE ERROR CHECKS
    014776 000137 015124        JMP   18$                  ;GO TO 18$ IF ERROR
615 015002      12$:
616
617      ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
618 015002 012737 000053 001412  MOV   #WCH!GO,RMCS10      ;WRITE CHECK COMMAND
619
620 015010 004737 036240      JSR   PC,PUT                ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    015014 000404                BR    13$                   ;GO TO 13$ IF NO ERROR
    015016 000240                NOP                    ;RETURN HERE IF ERROR
    015020 104000                EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
    015022 000137 015124        JMP   18$                  ;GO TO 18$ IF ERROR
621 015026      13$:
622
623      ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
624 015026 004737 036602      JSR   PC,TIMOUT            ;WAIT FOR READ TO COMPLETE
  
```

```

625
626 015032 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      015036 000404      BR     14$        ;GO TO 14$ IF NO ERROR
      015040 000240      NOP                    ;RETURN HERE IF ERROR
      015042 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
      015044 000137 015124      JMP    18$        ;GO TO 18$ IF ERROR
627 015050      14$:
628
629      ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
630 015050 004737 036766      JSR    PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      015054 000405      BR     15$        ;GO TO 15$ IF NO ERROR
      015056 000240      NOP                    ;RETURN HERE IF ERROR
      015060 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      015062 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      015064 000137 015124      JMP    18$        ;GO TO 18$ IF ERROR
631 015070      15$:
632 015070 004737 051502      JSR    PC,DTASTS   ;GO VERIFY RESULTS OF DATA TRANSFER
      015074 000405      BR     16$        ;GO TO 16$ IF NO ERROR
      015076 000240      NOP                    ;RETURN HERE IF ERROR
      015100 104000      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
      015102 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      015104 000137 015124      JMP    18$        ;GO TO 18$ IF ERROR
633 015110      16$:
634 015110 004737 037620      JSR    PC,SECERR   ;GO CHECK FOR SECONDARY ERRORS
      015114 000403      BR     17$        ;GO TO 17$ IF NO ERROR
      015116 000240      NOP                    ;RETURN HERE IF ERROR
      015120 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      015122 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
635 015124      17$:
636
637 015124      18$:
638
639
      ;*****
      ;*TEST 10      FORMAT CHECK ONES W/ WCE ERRORS
      ;*****
TST10:
      015124      SCOPE      ;SCOPE CALL
      015124 000004      NOP      ;START OF TEST
      015126 000240      MOV     #STACK,SP ;INITIALIZE STACK POINTER
      015130 012706 001100      MOV     $BASE,R0   ;R0 = UNIBUS ADDRESS
      015134 013700 001276      MOV     TSTQUE,R1  ;(R1) = DEVICE BEING TESTED
      015140 013701 001466      MOV     #10,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
      015144 012737 000010 001226
640
641      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
642 015152 012737 000000 001446      MOV     #0,RMDCO   ;CYLINDER = 0
643 015160 012737 000000 001420      MOV     #0,RMDAO   ;TRACK = 0, SECTOR 0
644 015166 012737 010000 001444      MOV     #FMT16,RMOFO ;16 BIT FORMAT
645 015174 012737 177376 001414      MOV     #-258,RMWCO ;2 + 256 WORDS (2'S COMP)
646 015202 012737 101360 001416      MOV     #BUFONE,RMBAO ;DATA BUFFER ADDRESS
647 015210 012737 000062 001412      MOV     #WH,RMCS10 ;WRITE HEADER AND DATA
648
649      ;VERIFY THAT SECTOR IS NOT BAD
      015216 004737 033152      JSR    PC,BADSCT   ;CALL BAD SECTOR MODULE
      015222 000405      BR     1$         ;GO TO 1$ IF NO ERROR
      015224 104401 063346      TYPE   ,SCTMSG     ;TYPE BAD SECTOR MESSAGE
      015230 100000      EMT                    ;ERROR # DEFINED BY BADSCT SUBROUTINE
      015232 000137 016062      JMP    20$        ;GO TO 20$ IF ERROR

```

```

650 015236
651 015236 012737 064736 001174
652 015244 012737 000001 001176
653 015252 004737 035104
654 015256
655
656
657 015256 004737 032176
    015262 154130
    015264 000404
    015266 000240
    015270 104000
    015272 000137 016062
658 015276
659
660
661 015276 012737 000005 001412
662 015304 012702 001553
663 015310 112722 000006
664 015314 112722 000034
665 015320 112722 000032
666 015324 112722 000000
667 015330 112722 000200
668
669 015334 004737 036240
    015340 000404
    015342 000240
    015344 104000
    015346 000137 016062
670 015352
671
672
673 015352 004737 035704
674 015356 004737 036602
675 015362
676
677
678 015362 004737 035770
    015366 000404
    015370 000240
    015372 104000
    015374 000137 016062
679 015400
680
681
682 015400 004737 044104
    015404 000405
    015406 000240
    015410 104000
    015412 004736

1$:
MOV #ONES,$TMP0 ;USE ALL ONES DATA PATTERN
MOV #1,$TMP1
JSR PC,GENBUF ;GO GENERATE DATA BUFFER

2$:
;PREPARE DEVICE FOR DATA TRANSFER
JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
.WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 3$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 20$ IF ERROR

BR 3$
NOP
EMT
JMP 20$

3$:
;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
MOV #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
MOVB #RMDA,(R2)+
MOVB #RMDC,(R2)+
MOVB #RMOF,(R2)+
MOVB #RMCS1,(R2)+
MOVB #200,(R2)+

JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
BR 4$ ;GO TO 4$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
EMT ;ERROR # DEFINED BY PUT SUBROUTINE
JMP 20$ ;GO TO 20$ IF ERROR

4$:
;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
JSR PC,GETSTS ;SETUP FOR STATUS
JSR PC,TIMOUT ;WAIT FOR SEEK TO COMPLETE

5$:
;GO READ SEEK STATUS
JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
BR 6$ ;GO TO 6$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
EMT ;ERROR # DEFINED BY GET SUBROUTINE
JMP 20$ ;GO TO 20$ IF ERROR

6$:
;VERIFY THE RESULTS OF THE SEEK COMMAND
JSR PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
BR 7$ ;GO TO 7$ IF NO ERROR
NOP ;RETURN HERE IF ERROR
EMT ;ERROR # DEFINED BY SEKSTS SUBROUTINE
JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
    
```

```

015414 000137 016062          JMP      20$          :GO TO 20$ IF ERROR
683 015420          7$:
684
685          :SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
686 015420 012737 000063 001412  MOV     #WCH!GO,RMCS10  :WRITE HEADER AND DATA
687 015420 012702 001556          MOV     #PUTINX+3,R2   :EXTEND REGISTER INDEX TABLE
688 015432 112722 000002          MOVB   #RMWC,(R2)+
689 015436 112722 000004          MOVB   #RMBIA,(R2)+
690 015442 112722 000000          MOVB   #RMCS1,(R2)+
691 015446 112722 000200          MOVB   #200,(R2)+
692
693 015452 004737 036240          JSR    PC,PUT          :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
        015456 000404          BR     8$             :GO TO 8$ IF NO ERROR
        015460 000240          NOP
        015462 104000          EMT
        015464 000137 016062          JMP    20$           :ERROR # DEFINED BY PUT SUBROUTINE
        015470          8$:
694          :WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
695          JSR    PC,TIMOUT  :WAIT FOR COMMAND TO COMPLETE
696
697 015470 004737 036602          JSR    PC,GET         :GO READ REGISTER(S) WITH GET SUBROUTINE
698          015500 000404          BR     9$             :GO TO 9$ IF NO ERROR
699          015502 000240          NOP
        015504 104000          EMT
        015506 000137 016062          JMP    20$           :ERROR # DEFINED BY GET SUBROUTINE
700          9$:
701
702          :VERIFY RESULTS OF WRITE COMMAND
703 015512 004737 036766          JSR    PC,PRIERR     :GO CHECK FOR PRIMARY ERRORS
        015516 000405          BR     10$            :GO TO 10$ IF NO ERROR
        015520 000240          NOP
        015522 104000          EMT
        015524 004736          JSR    PC,@(SP)+     :GO BACK FOR MORE ERROR CHECKS
        015526 000137 016062          JMP    20$           :GO TO 20$ IF ERROR
704          10$:
705          JSR    PC,DTASTS  :GO VERIFY RESULTS OF DATA TRANSFER
        015532 004737 051502          BR     11$            :GO TO 11$ IF NO ERROR
        015536 000405          NOP
        015540 000240          EMT
        015542 104000          EMT
        015544 004736          JSR    PC,@(SP)+     :GO BACK FOR MORE ERROR CHECKS
        015546 000137 016062          JMP    20$           :GO TO 20$ IF ERROR
706          11$:
707          JSR    PC,SECERR  :GO CHECK FOR SECONDARY ERRORS
        015552 004737 037620          BR     12$            :GO TO 12$ IF NO ERROR
        015556 000405          NOP
        015560 000240          EMT
        015562 104000          EMT
        015564 004736          JSR    PC,@(SP)+     :GO BACK FOR MORE ERROR CHECKS
        015566 000137 016062          JMP    20$           :GO TO 20$ IF ERROR
708          12$:
709
710          :ALTER DATA BUFFER
711 015572 005137 102362          COM    BUFTWO-?      :COMPLEMENT DATA WORD
712
713          :SETUP AND WRITE CHECK HEADER AND DATA COMMAND
714 015576 012737 000053 001412  MOV     #WCH!GO,RMCS10  :WRITE CHECK COMMAND
715
  
```

```

716 015604 004737 036240      JSR    PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      015610 000404      BR      13$        ;GO TO 13$ IF NO ERROR
      015612 000240      NOP                     ;RETURN HERE IF ERROR
      015614 104000      EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
      015616 000137 016062      JMP     20$        ;GO TO 20$ IF ERROR
717 015622      13$:
718
719      ;WAIT FOR WRITE (CHECK COMMAND TO COMPLETE AND READ STATUS
720 015622 004737 036602      JSR    PC,TIMOUT   ;WAIT FOR COMMAND TO COMPLETE
721
722 015626 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      015632 000404      BR      14$        ;GO TO 14$ IF NO ERROR
      015634 000240      NOP                     ;RETURN HERE IF ERROR
      015636 104000      EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
      015640 000137 016062      JMP     20$        ;GO TO 20$ IF ERROR
723 015644      14$:
724
725      ;CHECK FOR PRIMARY ERRORS
726 015644 004737 036766      JSR    PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      015650 000405      BR      15$        ;GO TO 15$ IF NO ERROR
      015652 000240      NOP                     ;RETURN HERE IF ERROR
      015654 104000      EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
      015656 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      015660 000137 016062      JMP     20$        ;GO TO 20$ IF ERROR
727 015664      15$:
728
729      ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
730 015664 032737 040000 001346  BIT     #WCE,RMCS2I ;IS WRITE CHECK ERROR SET??
731 015672 001022      BNE     17$        ;YES!!
732
733 015674 004737 051502      JSR    PC,DTASTS   ;GO VERIFY RESULTS OF DATA TRANSFER
      015700 000405      BR      16$        ;GO TO 16$ IF NO ERROR
      015702 000240      NOP                     ;RETURN HERE IF ERROR
      015704 104000      EMT                     ;ERROR # DEFINED BY DTASTS SUBROUTINE
      015706 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      015710 000137 016062      JMP     20$        ;GO TO 20$ IF ERROR
734 015714 013737 001346 001140 16$:  MOV     RMCS2I,$GDDAT ;LOAD EXPECTED STATUS
735 015722 052737 040000 001140      BIS     #WCE,$GDDAT
736 015730 013737 001346 001142      MOV     RMCS2I,$BDDAT ;LOAD RECEIVED STATUS
737 015736 104337      EMT     337
738 015740      17$:
739
740      ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
741 015740 012737 102362 001134      MOV     #BUFTWO-2,$GDADR ;LOAD EXPECTED ADDRESS
742 015746 013737 001342 001136      MOV     RMB1,$BDADR   ;LOAD RECEIVED ADDRESS
743 015754 162737 000002 001136      SUB     #2,$BDADR     ;DECREMENT RECEIVED ADDRESS
744
745      ;GET WCE DATA AND VERIFY IT IS OK
746 015762 112737 000022 001524      MOVB   #PMDB,GETINX  ;SETUP FOR READING RMDB
747 015770 112737 000200 001525      MOVB   #200,GETINX+1
748
749 015776 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      016002 000404      BR      18$        ;GO TO 18$ IF NO ERROR
      016004 000240      NOP                     ;RETURN HERE IF ERROR
      016006 104000      EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
      016010 000137 016062      JMP     20$        ;GO TO 20$ IF ERROR
750 016014 013737 001360 001142 18$:  MOV     RMDB1,$BDDAT ;LOAD RECEIVED DATA WORD
  
```



```

751 016022 013737 102362 001140      MOV      BUFTWO-2,$GDDAT ;LOAD EXPECTED DATA WORD
752 016030 005137 001140      COM      $GDDAT
753 016034 023737 001134 001136      CMP      $GDADR,$BDADR ;IS ADDRESS OK??
754 016042 001402                      BEQ      19$           ;YES!!
755 016044 104340                      EMT      340
756 016046 000405                      BR       20$
757 016050                      19$:
758 016050 023737 001140 001142      CMP      $GDDAT,$BDDAT ;IS DATA WORD OK??
759 016056 001401                      BEQ      20$           ;YES!!
760 016060 104341                      EMT      341
761 016062                      20$:
762
763
*****
*TEST 11          FORMAT MULTIPLE SECTORS
*****
TST11:
016062                      SCOPE                      ;SCOPE CALL
016062 000004                      NOP                        ;START OF TEST
016064 000240                      MOV      #STACK,SP        ;INITIALIZE STACK POINTER
016066 012706 001100                      MOV      $BASE,R0         ;R0 = UNIBUS ADDRESS
016072 013700 001276                      MOV      TSTQUE,R1        ;(R1) = DEVICE BEING TESTED
016076 013701 001466                      MOV      #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
016102 012737 000011 001226
764
765                      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
766 016110 012737 010000 001444      MOV      #FMT16,RMOFO    ;16 BIT FORMAT
767 016116 012737 000000 001446      MOV      #0,RMDCO        ;CYLINDER = 0
768 016124 012737 000000 001420      MOV      #0,RMDAO        ;TRACK = 0, SECTOR 0
769 016132 012737 176774 001414      MOV      #-258.*2,RMWCO  ;WORD COUNT FOR 2 SECTORS (2'S COMP)
770 016140 012737 101360 001416      MOV      #BUFONE,RMBAD   ;DATA BUFFER ADDRESS
771 016146 012737 000062 001412      MOV      #WH,RMCS10      ;WRITE HEADER AND DATA
772
773                      ;VERIFY THAT SECTOR IS NOT BAD
016154 004737 033152                      JSR      PC,BADSCT        ;CALL BAD SECTOR MODULE
016160 000405                      BR       1$              ;GO TO 1$ IF NO ERROR
016162 104401 063346                      TYPE     ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
016166 104000                      EMT
016170 000137 016656                      JMP      17$             ;GO TO 17$ IF ERROR
774 016174                      1$:
775 016174 012737 065000 001174      MOV      #ZEROS,$TMPO    ;USE ALL ZEROS DATA PATTERN
776 016202 012737 000001 001176      MOV      #1,$TMP1
777 016210 004737 035104                      JSR      PC,GENBUF       ;GO GENERATE DATA BUFFER
778 016214                      2$:
779
780                      ;PREPARE DEVICE FOR DATA TRANSFER
781 016214 004737 032176                      JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
016220 154130      .WORD 154130          ;TASK DESCRIPTOR AS FOLLOWS:
                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SE*
                                ;VERIFY RECALIBRATION
                                ;GO TO 3$ IF NO ERROR
                                ;RETURN HERE IF ERROR
016222 000404                      BR       3$
016224 000240                      NOP
016226 104000                      EMT
016230 000137 016656                      JMP      18$             ;GO TO 18$ IF ERROR

```

```

782 016234          3$:
783
784                ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON YLINDER
785 016234 012737 000005 001412  MOV      #SEEK!GO, RMCS10 ;CHANGE COMMAND TO SEEK
786 016242 012702 001553          MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
787 016246 112722 000006          MOVVB   #RMDA,(R2)+
788 016252 112722 000034          MOVVB   #RMDC,(R2)+
789 016256 112722 000032          MOVVB   #RMOF,(R2)+
790 016262 112722 000000          MOVVB   #RMCS1,(R2)+
791 016266 112722 000200          MOVVB   #200,(R2)+
792
793 016272 004737 036240          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      016276 000404          BR      4$            ;GO TO 4$ IF NO ERROR
      016300 000240          NOP
      016302 104000          EMT
      016304 000137 016656          JMP     18$           ;ERROR # DEFINED BY PUT SUBROUTINE
      016310
794                4$:
795                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
796                JSR     PC,GETSTS ;SETUP FOR STATUS
797 016310 004737 035704          JSR     PC,TIMOUT     ;WAIT FOR SEEK TO COMPLETE
798 016314 004737 036602
799 016320
800
801                ;GO READ SEEK STATUS
802 016320 004737 035770          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      016324 000404          BR      6$            ;GO TO 6$ IF NO ERROR
      016326 000240          NOP
      016330 104000          EMT
      016332 000137 016656          JMP     18$           ;ERROR # DEFINED BY GET SUBROUTINE
      016336
803                6$:
804                ;VERIFY THE RESULTS OF THE SEEK COMMAND
805                JSR     PC,SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
806 016336 004737 044104          BR      7$            ;GO TO 7$ IF NO ERROR
      016342 000405          NOP
      016344 000240          EMT
      016346 104000          JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      016350 004736          JMP     18$           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
      016352 000137 016656
807 016356          7$:
808
809                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
810 016356 012737 000063 001412  MOV      #WH!GO, RMCS10 ;WRITE HEADER AND DATA
811 016364 012702 001556          MOV      #PUTINX+3,R2 ;EXTEND REGISTER INDEX TABLE
812 016370 112722 000002          MOVVB   #RMWC,(R2)+
813 016374 112722 000004          MOVVB   #RMBA,(R2)+
814 016400 112722 000000          MOVVB   #RMCS1,(R2)+
815 016404 112722 000200          MOVVB   #200,(R2)+
816
817 016410 004737 036240          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      016414 000404          BR      8$            ;GO TO 8$ IF NO ERROR
      016416 000240          NOP
      016420 104000          EMT
      016422 000137 016656          JMP     18$           ;ERROR # DEFINED BY PUT SUBROUTINE
      016426
818                8$:
819                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
820                JSR     PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
821 016426 004737 036602
  
```

FORMAT MULTIPLE SECTORS

```

822
823 016432 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      016436 000404      BR     9$          ;GO TO 9$ IF NO ERROR
      016440 000240      NOP                    ;RETURN HERE IF ERROR
      016442 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
      016444 000137 016656      JMP    18$         ;GO TO 18$ IF ERROR
824 016450      9$:
825
826 ;VERIFY RESULTS OF WRITE COMMAND
827 016450 004737 036766      JSR    PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      016454 000405      BR     10$         ;GO TO 10$ IF NO ERROR
      016456 000240      NOP                    ;RETURN HERE IF ERROR
      016460 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      016462 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      016464 000137 016656      JMP    18$         ;GO TO 18$ IF ERROR
828 016470      10$:
829 016470 004737 051502      JSR    PC,DTASTS   ;GO VERIFY RESULTS OF DATA TRANSFER
      016474 000405      BR     11$         ;GO TO 11$ IF NO ERROR
      016476 000240      NOP                    ;RETURN HERE IF ERROR
      016500 104000      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
      016502 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      016504 000137 016656      JMP    18$         ;GO TO 18$ IF ERROR
830 016510      11$:
831 016510 004737 037620      JSR    PC,SECERR   ;GO CHECK FOR SECONDARY ERRORS
      016514 000405      BR     12$         ;GO TO 12$ IF NO ERROR
      016516 000240      NOP                    ;RETURN HERE IF ERROR
      016520 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      016522 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      016524 000137 016656      JMP    18$         ;GO TO 18$ IF ERROR
832 016530      12$:
833
834 ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
835 016530 012737 000053 001412  MOV    #WCH!GO,RMCS10 ;WRITE CHECK COMMAND
836
837 016536 004737 036240      JSR    PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      016542 000404      BR     13$         ;GO TO 13$ IF NO ERROR
      016544 000240      NOP                    ;RETURN HERE IF ERROR
      016546 104000      EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
      016550 000137 016656      JMP    18$         ;GO TO 18$ IF ERROR
838 016554      13$:
839
840 ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
841 016554 004737 036602      JSR    PC,TIMOUT   ;WAIT FOR WRITE CHECK TO FINISH
842
843 016560 004737 035770      JSR    PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      016564 000404      BR     14$         ;GO TO 14$ IF NO ERROR
      016566 000240      NOP                    ;RETURN HERE IF ERROR
      016570 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
      016572 000137 016656      JMP    18$         ;GO TO 18$ IF ERROR
844 016576      14$:
845
846 ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
847 016576 004737 036766      JSR    PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      016602 000405      BR     15$         ;GO TO 15$ IF NO ERROR
      016604 000240      NOP                    ;RETURN HERE IF ERROR
      016606 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      016610 004736      JSR    PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS

```

```

016612 000137 016656          JMP      18$          ;GO TO 18$ IF ERROR
848 016616          15$:          JSR      PC,DTASTS    ;GO VERIFY RESULTS OF DATA TRANSFER
849 016616 004737 051502          BR       16$          ;GO TO 16$ IF NO ERROR
      016622 000405          NOP          ;RETURN HERE IF ERROR
      016624 000240          EMT          ;ERROR # DEFINED BY DTASTS SUBROUTINE
      016626 104000          .SR      PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      016630 004736          JMP      18$          ;GO TO 18$ IF ERROR
850 016632 000137 016656          16$:          JSR      PC,SECERR    ;GO CHECK FOR SECONDARY ERRORS
851 016636 004737 037620          BR       17$          ;GO TO 17$ IF NO ERROR
      016642 000405          NOP          ;RETURN HERE IF ERROR
      016644 000240          EMT          ;ERROR # DEFINED BY SECERR SUBROUTINE
      016646 104000          .SR      PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      016650 004736          JMP      18$          ;GO TO 18$ IF ERROR
852 016652 000137 016656          17$:
853 016656          18$:
854 016656
855
856
;*****
;*TEST 12          FORMAT W/ HEAD SWITCHING
;*****
TST12:
      016656          SCOPE          ;SCOPE CALL
      016656 000004          NOP          ;START OF TEST
      016660 000240          MOV      #STACK,SP   ;INITIALIZE STACK POINTER
      016662 012706 001100          MOV      $BASE,R0    ;R0 = UNIBUS ADDRESS
      016666 013700 001276          MOV      TSTQUE,R1   ;(R1) = DEVICE BEING TESTED
      016672 013701 001466          MOV      #12,$TESTN  ;:SET TEST NUMBER IN APT MAIL BOX
      016676 012737 000012 001226
857
858          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
859 016704 012737 000000 001446          MOV      #0,RMDCO    ;CYLINDER = 0
860 016712 112737 000000 001421          MOV      #0,RMDAO+1  ;TRACK = 0
861 016720 112737 000037 001420          1$:          MOV      #31,RMDAO   ;SECTOR = 31
862 016726 012737 010000 001444          MOV      #FMT16,RMOFO ;16 BIT FORMAT
863 016734 012737 176774 001414          MOV      #-258,*2,RMWCO ;WORD COUNT FOR 2 SECTORS (2'S COMP)
864 016742 012737 101360 001416          MOV      #BUFONE,RMBAO ;DATA BUFFER ADDRESS
865 016750 012737 000062 001412          MOV      #WH,RMCSTO  ;WRITE HEADER AND DATA
866
867          ;VERIFY THAT SECTOR IS NOT BAD
      016756 004737 033152          JSR      PC,BADSCT   ;CALL BAD SECTOR MODULE
      016762 000405          BR       2$          ;GO TO 2$ IF NO ERROR
      016764 104401 063346          TYPE     ,SCTMSG     ;TYPE BAD SECTOR MESSAGE
      016770 104000          EMT          ;ERROR # DEFINED BY BADSCT SUBROUTINE
      016772 000137 017470          JMP      18$          ;GO TO 18$ IF ERROR
868 016776          2$:
869 016776 123727 001420 000037          CMPB     RMDAO,#31.  ;IS LAST SECTOR ASSIGNED ?
870 017004 001345          BNE     1$          ;BR IF NO
871 017006 012737 065000 001174          MOV      #ZEROS,$TMPO ;USE ALL ZEROS DATA PATTERN
872 017014 012737 000001 001176          MOV      #1,$TMP1
873 017022 004737 035104          JSR      PC,GENBUF   ;GO GENERATE DATA BUFFER
874
875          ;PREPARE DEVICE FOR DATA TRANSFER
876 017026 004737 032176          JSR      PC,TSTPRP  ;PREPARE DEVICE FOR TEST
      017032 154130          .WORD   154130      ;TASK DESCRIPTOR AS FOLLOWS:
      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
      ;CLEAR CONTROLLER & SELECT DEVICE

```

```

;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
;GO TO 3$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 18$ IF ERROR

017034 000404 BR 3$
017036 000240 NOP
017040 104000 EMT
017042 000137 017470 JMP 18$
877 017046 3$:
878
879 ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
880 017046 012737 000005 001412 MOV #SEEK!GO, RMCS10 ;CHANGE COMMAND TO SEEK
881 017054 012702 001553 MOV #PUTINX, R2 ;WRITE REGISTER INDEX TABLE
882 017060 112722 000006 MOVB #RMDA, (R2)+
883 017064 112722 000034 MOVB #RMDC, (R2)+
884 017070 112722 000032 MOVB #RMOF, (R2)+
885 017074 112722 000000 MOVB #RMCS1, (R2)+
886 017100 112722 000200 MOVB #200, (R2)+
887
888 017104 004737 036240 JSR PC, PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
017110 000404 BR 4$ ;GO TO 4$ IF NO ERROR
017112 000240 NOP ;RETURN HERE IF ERROR
017114 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
017116 000137 017470 JMP 18$ ;GO TO 18$ IF ERROR
889 017122 4$:
890
891 ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
892 017122 004737 035704 JSR PC, GETSTS ;SETUP FOR STATUS
893 017126 004737 036602 JSR PC, TIMEOUT ;WAIT FOR SEEK TO COMPLETE
894 017132 5$:
895
896 ;GO READ SEEK STATUS
897 017132 004737 035770 JSR PC, GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
017136 000404 BR 6$ ;GO TO 6$ IF NO ERROR
017140 000240 NOP ;RETURN HERE IF ERROR
017142 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
017144 000137 017470 JMP 18$ ;GO TO 18$ IF ERROR
898 017150 6$:
899
900 ;VERIFY THE RESULTS OF THE SEEK COMMAND
901 017150 004737 044104 JSR PC, SEKSTS ;GO VERIFY RESULTS OF SEEK OPERATION
017154 000405 BR 7$ ;GO TO 7$ IF NO ERROR
017156 000240 NOP ;RETURN HERE IF ERROR
017160 104000 EMT ;ERROR # DEFINED BY SEKSTS SUBROUTINE
017162 004736 JSR PC, @(SP)+ ;GO BACK FOR MORE ERROR CHECKS
017164 000137 017470 JMP 18$ ;GO TO 18$ IF ERROR
902 017170 7$:
903
904 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
905 017170 012737 000063 001412 MOV #WH.GO, RMCS10 ;WRITE HEADER AND DATA
906 017176 012702 001556 MOV #PUTINX+3, R2 ;EXTEND REGISTER INDEX TABLE
907 017202 112722 000002 MOVB #RMWC, (R2)+
908 017206 112722 000004 MOVB #RMSA, (R2)+
909 017212 112722 000000 MOVB #RMCS1, (R2)+
910 017216 112722 000200 MOVB #200, (R2)+
911

```



```

939 017404 000137 017470          JMP      18$          ;GO TO 18$ IF ERROR
940 017410          14$:
941          ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
942 017410 004737 036766          JSR      PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
          017414 000405          BR       15$          ;GO TO 15$ IF NO ERROR
          017416 000240          NOP
          017420 104000          EMT
          017422 004736          JSR      PC,@(SP)+   ;ERROR # DEFINED BY PRIERR SUBROUTINE
          017424 000137 017470          JMP      18$          ;GO BACK FOR MORE ERROR CHECKS
943 017430          15$:
944 017430 004737 051502          JSR      PC,DTASTS   ;GO VERIFY RESULTS OF DATA TRANSFER
          017434 000405          BR       16$          ;GO TO 16$ IF NO ERROR
          017436 000240          NOP
          017440 104000          EMT
          017442 004736          JSR      PC,@(SP)+   ;ERROR # DEFINED BY DTASTS SUBROUTINE
          017444 000137 017470          JMP      18$          ;GO BACK FOR MORE ERROR CHECKS
945 017450          16$:
946 017450 004737 037620          JSR      PC,SECERR   ;GO CHECK FOR SECONDARY ERRORS
          017454 000405          BR       17$          ;GO TO 17$ IF NO ERROR
          017456 000240          NOP
          017460 104000          EMT
          017462 004736          JSR      PC,@(SP)+   ;ERROR # DEFINED BY SECERR SUBROUTINE
          017464 000137 017470          JMP      18$          ;GO BACK FOR MORE ERROR CHECKS
947 017470          17$:
948          18$:
949 017470
950
951

```

```

*****
;*TEST 13      FORMAT W/ MID TRANSFER SEEK
*****
TST13:

```

```

          017470
          017470 000004
          017472 000240
          017474 012706 001100
          017500 013700 001276
          017504 013701 001466
          017510 012737 000013 001226
          SCOPE          ;SCOPE CALL
          NOP           ;START OF TEST
          MOV      #STACK,SP ;INITIALIZE STACK POINTER
          MOV      $BASE,R0  ;R0 = UNIBUS ADDRESS
          MOV      TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
          MOV      #13,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
952
953
954 017516 012737 000000 001446 ;:SETUP PARAMETERS FOR GENERATING DATA BUFFER
955 017524 013737 001334 001420 1$: MOV      #0,RMDCO      ;CYLINDER = 0
956 017532 112737 000037 001420 MOV      LSTRK,RMDAO   ;START LAST TRACK AND
957 017540 012737 010000 001444 MOVVB   #31.,RMDAO     ;LAST SECTOR
958 017546 012737 176774 001414 MOV      #FMT16,RMOFO  ;16 BIT FORMAT
959 017554 012737 101360 001416 MOV      #-258.*2,RMWCO ;WORD COUNT FOR 2 SECTORS (2'S COMP)
960 017562 012737 000062 001412 MOV      #BUFONE,RMBAO ;DATA BUFFER ADDRESS
961          MOV      #WH,RMC$10 ;WRITE HEADER AND DATA
962
          ;VERIFY THAT SECTOR IS NOT BAD
          017570 004737 033152          JSR      PC,BADSCT   ;CALL BAD SECTOR MODULE
          017574 000405          BR       2$          ;GO TO 2$ IF NO ERROR
          017576 104401 063346          TYPE    ,SCTMSC     ;TYPE BAD SECTOR MESSAGE
          017602 104000          EMT
          017604 000137 020302          JMP      19$          ;ERROR # DEFINED BY BADSCT SUBROUTINE
963 017610          2$:
964 017610 123737 001421 001335          CMPB   RMDAO+1,LSTRK+1 ;IS LAST TRACK ASSIGNED ?
965 017616 001342          BNE    1$          ;BR IF NO

```

```

966 017620 012737 065000 001174      MOV      #ZEROS,$TMP0      ;USE ALL ZEROS DATA PATTERN
967 017626 012737 000001 001176      MOV      #1,$TMP1
968 017634 004737 035104              JSR      PC,GENBUF        ;GO GENERATE DATA BUFFER
969 017640
970
971
972 017640 004737 032176              ;PREPARE DEVICE FOR DATA TRANSFER
    017644 154130              JSR      PC,TSTPRP      ;PREPARE DEVICE FOR TEST
                                .WORD      154130      ;TASK DESCRIPTOR AS FOLLOWS:
                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 4$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 19$ IF ERROR
    017646 000404              BR       4$
    017650 000240              NOP
    017652 104000              EMT
    017654 000137 020302          JMP      19$
973 017660              4$:
974
975
976 017660 012737 000005 001412          ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
    017666 012702 001553          MOV      #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SFEK
    017672 112722 000006          MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
    017676 112722 000034          MOV      #RMDA,(R2)+
    017702 112722 000032          MOV      #RMDC,(R2)+
    017706 112722 000000          MOV      #RMOF,(R2)+
    017712 112722 000200          MOV      #RMCS1,(R2)+
    017716 004737 036240          MOV      #200,(R2)+
    017722 000404              JSR      PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    017724 000240              BR       5$
    017726 104000              NOP
    017730 000137 020302          EMT
    017734              JMP      19$          ;GO TO 19$ IF ERROR
985
986
987
988 017734 004737 035704          ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
    017740 004737 036602          JSR      PC,GETSTS      ;SETUP FOR STATUS
    017744              JSR      PC,TIMQUT      ;WAIT FOR SEEK TO COMPLETE
989
990
991
992
993 017744 004737 035770          ;GO READ SEEK STATUS
    017750 000404              JSR      PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
    017752 000240              BR       7$
    017754 104000              NOP
    017756 000137 020302          EMT
    017762              JMP      19$          ;GO TO 19$ IF ERROR
994
995
996
997 017762 004737 044104          ;VERIFY THE RESULTS OF THE SEEK COMMAND
    017766 000405              JSR      PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
    017770 000240              BR       8$
    017772 104000              NOP
    017774 004736              EMT
    017776 000137 020302          JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
                                JMP      19$          ;GO TO 19$ IF ERROR

```



```

998 020002      8$:
999
1000           ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1001 020002 012737 000063 001,12  MOV    #WH!GO, RMCS10 ;WRITE HEADER AND DATA
1002 020010 012702 001556          MOV    #PUTINX+3, R2 ;EXTEND REGISTER INDEX TABLE
1003 020014 112722 000002          MOVB  #RMWC, (R2)+
1004 020020 112722 000004          MOVB  #RMBA, (R2)+
1005 020024 112722 000000          MOVB  #RMCS1, (R2)+
1006 020030 112722 000200          MOVB  #200, (R2)+
1007
1008 020034 004737 036240          JSR    PC, PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      020040 000404          BR     9$ ;GO TO 9$ IF NO ERROR
      020042 000240          NOP ;RETURN HERE IF ERROR
      020044 104000          EMT ;ERROR # DEFINED BY PUT SUBROUTINE
      020046 000137 020302      JMP    19$ ;GO TO 19$ IF ERROR
1009 020052      9$:
1010
1011           ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
1012 020052 004737 036602          JSR    PC, TIMEOUT ;WAIT FOR COMMAND TO COMPLETE
1013
1014 020056 004737 035770          JSR    PC, GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
      020062 000404          BR     10$ ;GO TO 10$ IF NO ERROR
      020064 000240          NOP ;RETURN HERE IF ERROR
      020066 104000          EMT ;ERROR # DEFINED BY GET SUBROUTINE
      020070 000137 020302      JMP    19$ ;GO TO 19$ IF ERROR
1015 020074
1016
1017           ;VERIFY RESULTS OF WRITE COMMAND
1018 020074 004737 036766          JSR    PC, PRIERR ;GO CHECK FOR PRIMARY ERRORS
      020100 000405          BR     11$ ;GO TO 11$ IF NO ERROR
      020102 000240          NOP ;RETURN HERE IF ERROR
      020104 104000          EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
      020106 004736          JSR    PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      020110 000137 020302      JMP    19$ ;GO TO 19$ IF ERROR
1019 020114
1020 020114 004737 051502          JSR    PC, DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      020120 000405          BR     12$ ;GO TO 12$ IF NO ERROR
      020122 000240          NOP ;RETURN HERE IF ERROR
      020124 104000          EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
      020126 004736          JSR    PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      020130 000137 020302      JMP    19$ ;GO TO 19$ IF ERROR
1021 020134
1022 020134 004737 037620          JSR    PC, SECERR ;GO CHECK FOR SECONDARY ERRORS
      020140 000405          BR     13$ ;GO TO 13$ IF NO ERROR
      020142 000240          NOP ;RETURN HERE IF ERROR
      020144 104000          EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
      020146 004736          JSR    PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      020150 000137 020302      JMP    19$ ;GO TO 19$ IF ERROR
1023 020154
1024
1025           ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
1026 020154 012737 000053 001412  MOV    #WCH!GO, RMCS10 ;WRITE CHECK COMMAND
1027
1028 020162 004737 036240          JSR    PC, PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      020166 000404          BR     14$ ;GO TO 14$ IF NO ERROR
      020170 000240          NOP ;RETURN HERE IF ERROR
      020172 104000          EMT ;ERROR # DEFINED BY PUT SUBROUTINE

```

```

1029 020174 00137 020302      JMP      19$      :GO TO 19$ IF ERROR
1030 020200
1031 ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
1032 020200 004737 036602      JSR      PC,TIMOUT :WAIT FOR WRITE CHECK TO FINISH
1033
1034 020204 004737 035770      JSR      PC,GET    :GO READ REGISTER(S) WITH GET SUBROUTINE
      020210 000404      BR       15$      :GO TO 15$ IF NO ERROR
      020212 000240      NOP
      020214 104000      EMT
      020216 000137 020302      JMP      19$      :ERROR # DEFINED BY GET SUBROUTINE
      020222
1035 15$:
1036
1037 ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
1038 020222 004737 036766      JSR      PC,PRIERR :GO CHECK FOR PRIMARY ERRORS
      020226 000405      BR       16$      :GO TO 16$ IF NO ERROR
      020230 000240      NOP
      020232 104000      EMT
      020234 004736      JSR      PC,@(SP)+ :ERROR # DEFINED BY PRIERR SUBROUTINE
      020236 000137 020302      JMP      19$      :GO BACK FOR MORE ERROR CHECKS
      :GO TO 19$ IF ERROR
1039 16$:
1040 020242 004737 051502      JSR      PC,DTASTS :GO VERIFY RESULTS OF DATA TRANSFER
      020246 000405      BR       17$      :GO TO 17$ IF NO ERROR
      020250 000240      NOP
      020252 104000      EMT
      020254 004736      JSR      PC,@(SP)+ :ERROR # DEFINED BY DTASTS SUBROUTINE
      020256 000137 020302      JMP      19$      :GO BACK FOR MORE ERROR CHECKS
      :GO TO 19$ IF ERROR
1041 17$:
1042 020262 004737 037620      JSR      PC,SECERR :GO CHECK FOR SECONDARY ERRORS
      020266 000405      BR       18$      :GO TO 18$ IF NO ERROR
      020270 000240      NOP
      020272 104000      EMT
      020274 004736      JSR      PC,@(SP)+ :ERROR # DEFINED BY SECERR SUBROUTINE
      020276 000137 020302      JMP      19$      :GO BACK FOR MORE ERROR CHECKS
      :GO TO 19$ IF ERROR
1043 18$:
1044
1045 19$:
1046
1047
:*****
:*TEST 14 ' FORMAT W/ IMPLIED SEEK
:*****
TST14:
      SCOPE      :SCOPE CALL
      NOP
      MOV #STACK,SP :START OF TEST
      MOV $BASE,R0  :INITIALIZE STACK POINTER
      MOV TSTQUE,R1 :R0 = UNIBUS ADDRESS
      MOV #14,$TESTN : (R1) = DEVICE BEING TESTED
      :SET TEST NUMBER IN APT MAIL BOX

1048 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
1049
1050 MOV #0,RMDCO :CYLINDER = 0
1051 MOV #0,RMDAO :TRACK = 0, SECTOR = 0
1052 MOV #FMT16,RMOFO :16 BIT FORMAT
1053 MOV #-258.*2,RMWCO :WORD COUNT FOR 2 SECTORS (2'S LOMP)
1054 MOV #BUFONE,RMBAO :DATA BUFFER ADDRESS
1055 MOV #WH,RMCS10 :WRITE HEADER AND DATA
1056

```

```

1057      020374 004737 033152      ;VERIFY THAT SECTOR IS NOT BAD
          020400 000405                JSR    PC,BADSECT      ;CALL BAD SECTOR MODULE
          020402 104401 063346        BR     1$              ;GO TO 1$ IF NO ERROR
          020406 104000                TYPE   ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
          020410 000137 021120        EMT                    ;ERROR # DEFINED BY BADSECT SUBROUTINE
          020414 000137 021120        JMP    18$            ;GO TO 18$ IF ERROR
1058      020414 012737 065000 001174 1$:      MOV    #ZEROS,$TMP0    ;USE ALL ZEROS DATA PATTERN
1059      020414 012737 000001 001176        MOV    #1,$TMP1
1060      020422 012737 000001 001176        JSR    PC,GENBUF      ;GO GENERATE DATA BUFFER
1061      020430 004737 035104
1062      020434
1063
1064
1065      020434 004737 032176      ;PREPARE DEVICE FOR DATA TRANSFER
          020440 154130                JSR    PC,TSTPRP      ;PREPARE DEVICE FOR TEST
          .WORD 154130                  .WORD 154130          ;TASK DESCRIPTOR AS FOLLOWS:
          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
          ;CLEAR CONTROLLER & SELECT DEVICE
          ;VERIFY CONTROLLER CLEAR OPERATION
          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
          ;VERIFY PACK ACKNOWLEDGE
          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
          ;VERIFY RECALIBRATION
          020442 000404                BR     3$              ;GO TO 3$ IF NO ERROR
          020444 000240                NOP
          020446 104000                EMT                    ;RETURN HERE IF ERROR
          020450 000137 021120        JMP    18$            ;ERROR # DEFINED BY TSTPRP SUBROUTINE
          020454 000137 021120        JMP    18$            ;GO TO 18$ IF ERROR
1066      020454
1067
1068
1069      020454 013737 001446 021122      ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE OFF CYLINDER
          020462 012737 001466 001446        MOV    RMDCO,19$      ;SAVE CYLINDER ADDRESS
          020470 012737 000005 001412        MOV    #822.,RMDCO    ;SEEK TO LAST CYLINDER
          020476 012702 001553                MOV    #SEEK!GO,RMCS10 ;CHANGE COMMAND TO SEEK
          020502 112722 000006                MOV    #PUTINX,R2     ;WRITE REGISTER INDEX TABLE
          020506 112722 000034                MOV    #RMDA,(R2)+
          020512 112722 000032                MOV    #RMDC,(R2)+
          020516 112722 000000                MOV    #RMOF,(R2)+
          020522 112722 000200                MOV    #RMCS1,(R2)+
          020526 004737 036240                MOV    #200,(R2)+
          020532 000404                JSR    PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
          020534 000240                BR     4$              ;GO TO 4$ IF NO ERROR
          020536 104000                NOP                    ;RETURN HERE IF ERROR
          020540 000137 021120        FMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
          020544 000137 021120        JMP    18$            ;GO TO 18$ IF ERROR
1080      020544
1081
1082
1083      020544 004737 035704      ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
          020550 004737 036602                JSR    PC,GETSTS      ;SETUP FOR STATUS
          020554 000137 021120                JSR    PC,TIMOUT     ;WAIT FOR SEEK TO COMPLETE
1085      020554
1086
1087
1088      020554 004737 035770      ;GO READ SEEK STATUS
          020560 000404                JSR    PC,GET         ;GO READ REGISTER(S) WITH GET SUBROUTINE
          020562 000240                BR     6$              ;GO TO 6$ IF NO ERROR
          020564 104000                NOP                    ;RETURN HERE IF ERROR
          020566 000137 021120        EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
          020566 000137 021120        JMP    18$            ;GO TO 18$ IF ERROR
  
```

```

1089 020572          6$:
1090
1091          ;VERIFY THE RESULTS OF THE SEEK COMMAND
1092 020572 00473/ 044104      JSR    PC,SEKSTS      ;GO VERIFY RESULTS OF SEEK OPERATION
          020576 000405      BR     7$            ;GO TO 7$ IF NO ERROR
          020600 000240      NOP                    ;RETURN HERE IF ERROR
          020602 104000      EMT                    ;ERROR # DEFINED BY SEKSTS SUBROUTINE
          020604 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
          020606 000137 021120      JMP    18$           ;GO TO 18$ IF ERROR
1093 020612
1094
1095          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1096 020612 013737 021122 001446      MOV    19$,RMDCO      ;RESTORE DISK ADDRESS
1097 020620 012737 000063 001412      MOV    #WH!GO,RMCS10 ;WRITE HEADER AND DATA
1098 020626 012702 001556              MOV    #PUTINX+3,R2  ;EXTEND REGISTER INDEX TABLE
1099 020632 112722 000002              MOVB  #RMWC,(R2)+
1100 020636 112722 000004              MOVB  #RMB3A,(R2)+
1101 020642 112722 000000              MOVB  #RMCS1,(R2)+
1102 020646 112722 000200              MOVB  #200,(R2)+
1103
1104 020652 004737 036240      JSR    PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
          020656 000404      BR     8$            ;GO TO 8$ IF NO ERROR
          020660 000240      NOP                    ;RETURN HERE IF ERROR
          020662 104000      EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
          020664 000137 021120      JMP    18$           ;GO TO 18$ IF ERROR
1105 020670
1106
1107          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
1108 020670 004737 036602      JSR    PC,TIMOUT     ;WAIT FOR COMMAND TO COMPLETE
1109
1110 020674 004737 035770      JSR    PC,GET        ;GO READ REGISTER(S) WITH GET SUBROUTINE
          020700 000404      BR     9$            ;GO TO 9$ IF NO ERROR
          020702 000240      NOP                    ;RETURN HERE IF ERROR
          020704 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
          020706 000137 021120      JMP    18$           ;GO TO 18$ IF ERROR
1111 020712
1112
1113          ;VERIFY RESULTS OF WRITE COMMAND
1114 020712 004737 036766      JSR    PC,PRIERR     ;GO CHECK FOR PRIMARY ERRORS
          020716 000405      BR     10$           ;GO TO 10$ IF NO ERROR
          020720 000240      NOP                    ;RETURN HERE IF ERROR
          020722 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
          020724 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
          020726 000137 021120      JMP    18$           ;GO TO 18$ IF ERROR
1115 020732
1116 020732 004737 051502      10$:      JSR    PC,DTASTS     ;GO VERIFY RESULTS OF DATA TRANSFER
          020736 000405      BR     11$           ;GO TO 11$ IF NO ERROR
          020740 000240      NOP                    ;RETURN HERE IF ERROR
          020742 104000      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
          020744 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
          020746 000137 021120      JMP    18$           ;GO TO 18$ IF ERROR
1117 020752
1118 020752 004737 037620      11$:      JSR    PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
          020756 000405      BR     12$           ;GO TO 12$ IF NO ERROR
          020760 000240      NOP                    ;RETURN HERE IF ERROR
          020762 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
          020764 004736      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
    
```

```

1119 020766 000137 021120      JMP      18$      ;GO TO 18$ IF ERROR
1120 020772
1121
1122 020772 012737 000053 001412 ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
1123      MOV      #WCH!GO, RMCS10 ;WRITE CHECK COMMAND
1124 021000 004737 036240      JSR      PC, PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      021004 000404      BR      13$      ;GO TO 13$ IF NO ERROR
      021006 000240      NOP
      021010 104000      EMT      ;RETURN HERE IF ERROR
      021012 000137 021120      JMP      18$      ;ERROR # DEFINED BY PUT SUBROUTINE
      ;GO TO 18$ IF ERROR
1125 021016
1126
1127      ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
1128 021016 004737 036602      JSR      PC, TIMEOUT ;WAIT FOR WRITE CHECK TO FINISH
1129
1130 021022 004737 035770      JSR      PC, GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      021026 000404      BR      14$      ;GO TO 14$ IF NO ERROR
      021030 000240      NOP      ;RETURN HERE IF ERROR
      021032 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      021034 000137 021120      JMP      18$      ;GO TO 18$ IF ERROR
1131 021040
1132
1133      ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
1134 021040 004737 036766      JSR      PC, PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      021044 000405      BR      15$      ;GO TO 15$ IF NO ERROR
      021046 000240      NOP      ;RETURN HERE IF ERROR
      021050 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      021052 004736      JSR      PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      021054 000137 021120      JMP      18$      ;GO TO 18$ IF ERROR
1135 021060
1136 021060 004737 051502      JSR      PC, DTASTS   ;GO VERIFY RESULTS OF DATA TRANSFER
      021064 000405      BR      16$      ;GO TO 16$ IF NO ERROR
      021066 000240      NOP      ;RETURN HERE IF ERROR
      021070 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      021072 004736      JSR      PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      021074 000137 021120      JMP      18$      ;GO TO 18$ IF ERROR
1137 021100
1138 021100 004737 037620      JSR      PC, SECERR   ;GO CHECK FOR SECONDARY ERRORS
      021104 000405      BR      17$      ;GO TO 17$ IF NO ERROR
      021106 000240      NOP      ;RETURN HERE IF ERROR
      021110 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      021112 004736      JSR      PC, @ (SP)+ ;GO BACK FOR MORE ERROR CHECKS
      021114 000137 021120      JMP      18$      ;GO TO 18$ IF ERROR
1139 021120
1140
1141 021120 000401      18$: BR      20$
1142
1143 021122 000000      19$: .WORD 0      ;TEMPORARY STORAGE
1144
1145 021124      20$:
1146
1147
;*****
;*TEST 15      FORMAT EACH SECTOR ADDRESS
;*****
TST15:
      021124
      021124 000004      SCOPE      ;SCOPE CALL

```



```

1181 021354 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
1182 021356 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR
1183 021362 ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1184 021362 004737 035704 JSR PC,GETSTS
1185 021366 004737 036602 JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
1186 021372 004737 035770 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
1187 021376 000404 BR 6$ ;GO TO 6$ IF NO ERROR
021400 000240 NOP ;RETURN HERE IF ERROR
021402 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
021404 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR
1188 021410 6$:
1189
1190 ;VERIFY RESULTS OF WRITE COMMAND
1191 021410 004737 036766 JSR PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
021414 000405 BR 7$ ;GO TO 7$ IF NO ERROR
021416 000240 NOP ;RETURN HERE IF ERROR
021420 104000 EMT ;ERROR # DEFINED BY PRIERR SUBROUTINE
021422 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
021424 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR
1192 021430 7$:
1193 021430 004737 051502 JSR PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
021434 000405 BR 8$ ;GO TO 8$ IF NO ERROR
021436 000240 NOP ;RETURN HERE IF ERROR
021440 104000 EMT ;ERROR # DEFINED BY DTASTS SUBROUTINE
021442 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
021444 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR
1194 021450 8$:
1195 021450 004737 037620 JSR PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
021454 000405 BR 9$ ;GO TO 9$ IF NO ERROR
021456 000240 NOP ;RETURN HERE IF ERROR
021460 104000 EMT ;ERROR # DEFINED BY SECERR SUBROUTINE
021462 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
021464 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR
1196 021470 9$:
1197
1198 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
1199 021470 012737 000073 001412 MOV #RH!GO,RMC$10 ;READ HEADER & DATA COMMAND
1200 021476 012737 102364 001416 MOV #BUFTWO,RMBAG ;CHANGE BUS ADDRESS
1201
1202 021504 004737 036240 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
021510 000404 BR 10$ ;GO TO 10$ IF NO ERROR
021512 000240 NOP ;RETURN HERE IF ERROR
021514 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
021516 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR
1203 021522 10$:
1204
1205 ;WAIT FOR READ TO COMPLETE AND GET STATUS
1206 021522 004737 036602 JSR PC,TIMOUT ;WAIT FOR READ TO COMPLETE
1207
1208 021526 004737 035770 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
021532 000404 BR 11$ ;GO TO 11$ IF NO ERROR
021534 000240 NOP ;RETURN HERE IF ERROR
021536 104000 EMT ;ERROR # DEFINED BY GET SUBROUTINE
021540 000137 021666 JMP 16$ ;GO TO 16$ IF ERROR

```

```

1209 021544      11$:
1210
1211
1212 021544 004737 036766      ;VERIFY THE RESULTS OF READ OPERATION
      JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      BR     12$            ;GO TO 12$ IF NO ERROR
      NOP                    ;RETURN HERE IF ERROR
      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      JMP    16$            ;GO TO 16$ IF ERROR
1213 021564
1214 021564 004737 051502      12$:
      JSR    PC,DTASTS     ;GO VERIFY RESULTS OF DATA TRANSFER
      BR     13$            ;GO TO 13$ IF NO ERROR
      NOP                    ;RETURN HERE IF ERROR
      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      JMP    16$            ;GO TO 16$ IF ERROR
1215 021604
1216 021604 004737 037620      13$:
      JSR    PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
      BR     14$            ;GO TO 14$ IF NO ERROR
      NOP                    ;RETURN HERE IF ERROR
      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      JSR    PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      JMP    16$            ;GO TO 16$ IF ERROR
1217 021624
1218
1219
1220 021624 004737 035342      14$:
      ;VERIFY DATA
      JSR    PC,CMPBUF     ;GO COMPARE WRITE, READ DATA BUFFERS
      .WORD  BUFONE        ;STARTING ADDRESS OF WRITE BUFFER
      .WORD  BJFTWO        ;STARTING ADDRESS OF READ BUFFER
      BR     15$            ;GO TO 15$ IF NO ERROR
      NOP                    ;RETURN HERE IF ERROR
      EMT                    ;ERROR # DEFINED BY CMPBUF SUBROUTINE
      JMP    16$            ;GO TO 16$ IF ERROR
1221
1222
1223 021646
1224 021646 005237 001420      ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
1225 021652 122737 000037 001420      15$:
      INC    RMDAO          ;ADVANCE SECTOR COUNT
      CMPB  #31.,RMDAO     ;DONE ALL SECTORS??
      BLO   16$            ;YES!!
      JMP   1$              ;GO DO NEXT SECTOR
1226 021660 103402
1227 021662 000137 021174
1228 021666
1229
1230
      ;*****
      ;*TEST 16      FORMAT EACH TRACK ADDRESS
      ;*****
      TST16:
      SCOPE                    ;SCOPE CALL
      NOP                      ;START OF TEST
      MOV    #STACK,SP        ;INITIALIZE STACK POINTER
      MOV    $BASE,R0         ;R0 - UNIBUS ADDRESS
      MOV    TSTQUE,R1        ;:(R1) = DEVICE BEING TESTED
      MOV    #16,$TESTN       ;:SET TEST NUMBER IN APT MAIL BOX
1231
1232
1233 021714 012737 000000 001446      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
1234 021722 012737 000000 001420
1235 021730 012737 010000 001444
      MOV    #0,RMDCQ         ;CYLINDER = 0
      MOV    #0,RMDAO        ;TRACK = 0, SECTOR = 0
      MOV    #FMT16,RMFO     ;16 BIT FORMAT
  
```



```

1236 021736 1$:
1237 021736 012737 177376 001414 MOV #-258.,RMWCO ;2 + 256 WORDS (2'S COMP)
1238 021744 012737 101360 001416 MOV #BUFONE,RMBAO ;DATA BUFFER ADDRESS
1239 021752 012737 000062 001412 MOV #WH,RMCS10 ;WRITE HEADER AND DATA
1240
1241 ;VERIFY THAT SECTOR IS NOT BAD
021760 004737 033152 JSR PC,BADSCT ;CALL BAD SECTOR MODULE
021764 000405 BR 2$ ;GO TO 2$ IF NO ERROR
021766 104401 063346 TYPE ,SCTMSG ;TYPE BAD SECTOR MESSAGE
021772 104000 EMT ;ERROR # DEFINED BY BADSCT SUBROUTINE
021774 000137 022430 JMP 16$ ;GO TO 16$ IF ERROR
1242 022000 2$:
1243 022000 012737 001420 001174 MOV #RMDAO,$TMP0 ;USE TRACK FOR DATA PATTERN
1244 022006 012737 000001 001176 MOV #1,$TMP1
1245 022014 004737 035104 JSR PC,GENBUF ;GO GENERATE DATA BUFFER
1246 022020 3$:
1247 ;PREPARE DEVICE FOR DATA TRANSFER
1248 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
1249 022020 004737 032176 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
022024 154130 ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
;VERIFY RECALIBRATION
022026 000404 BR 4$ ;GO TO 4$ IF NO ERROR
022030 000240 NOP ;RETURN HERE IF ERROR
022032 104000 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
022034 000137 022430 JMP 16$ ;GO TO 16$ IF ERROR
1250 022040 4$:
1251 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1252 MOV #WH!GO,RMCS10 ;WRITE HEADER AND DATA
1253 022040 012737 000063 001412 MOV #PUTINX,R2 ;WRITE REGISTER INDEX TABLE
1254 022046 012702 001553 MOV #RMDA,(R2)+
1255 022052 112722 000006 MOV #RMDC,(R2)+
1256 022056 112722 000034 MOV #RMOF,(R2)+
1257 022062 112722 000032 MOV #RMWC,(R2)+
1258 022066 112722 000002 MOV #RMBA,(R2)+
1259 022072 112722 000004 MOV #RMCS1,(R2)+
1260 022076 112722 000000 MOV #200,(R2)+
1261 022102 112722 000200
1262
1263 022106 004737 036240 JSR PC,PUT ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
022112 000404 BR 5$ ;GO TO 5$ IF NO ERROR
022114 000240 NOP ;RETURN HERE IF ERROR
022116 104000 EMT ;ERROR # DEFINED BY PUT SUBROUTINE
022120 000137 022430 JMP 16$ ;GO TO 16$ IF ERROR
1264 022124 5$:
1265 ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1266 JSR PC,GETSTS
1267 022124 004737 035704 JSR PC,TIMOUT ;WAIT FOR COMMAND TO COMPLETE
1268 022130 004737 036602
1269
1270 022134 004737 035770 JSR PC,GET ;GO READ REGISTER(S) WITH GET SUBROUTINE
022140 000404 BR 6$ ;GO TO 6$ IF NO ERROR
  
```

	022142	000240			NOP				:RETURN HERE IF ERROR
	022144	104000			EMT				:ERROR # DEFINED BY GET SUBROUTINE
	022146	000137	022430		JMP	16\$:GO TO 16\$ IF ERROR
1271	022152						6\$:		
1272									
1273									
1274	022152	004737	036766						:VERIFY RESULTS OF WRITE COMMAND
	022156	000405			JSR	PC,PRIERR			:GO CHECK FOR PRIMARY ERRORS
	022160	000240			BR	7\$:GO TO 7\$ IF NO ERROR
	022162	104000			NOP				:RETURN HERE IF ERROR
	022164	004736			EMT				:ERROR # DEFINED BY PRIERR SUBROUTINE
	022166	000137	022430		JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
1275	022172				JMP	16\$:GO TO 16\$ IF ERROR
1276	022172	004737	051502				7\$:		
	022176	000405			JSR	PC,DTASTS			:GO VERIFY RESULTS OF DATA TRANSFER
	022200	000240			BR	8\$:GO TO 8\$ IF NO ERROR
	022202	104000			NOP				:RETURN HERE IF ERROR
	022204	004736			EMT				:ERROR # DEFINED BY DTASTS SUBROUTINE
	022206	000137	022430		JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
1277	022212				JMP	16\$:GO TO 16\$ IF ERROR
1278	022212	004737	037620				8\$:		
	022216	000405			JSR	PC,SECERR			:GO CHECK FOR SECONDARY ERRORS
	022220	000240			BR	9\$:GO TO 9\$ IF NO ERROR
	022222	104000			NOP				:RETURN HERE IF ERROR
	022224	004736			EMT				:ERROR # DEFINED BY SECERR SUBROUTINE
	022226	000137	022430		JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
1279	022232				JMP	16\$:GO TO 16\$ IF ERROR
1280							9\$:		
1281									:READ HEADER AND DATA FOR SECTOR JUST WRITTEN
1282	022232	012737	000073	001412	MOV	#RH.GO,RMCS10			:READ HEADER & DATA COMMAND
1283	022240	012737	102364	001416	MOV	#BUFTWO,RMBAO			:CHANGE BUS ADDRESS
1284									
1285	022246	004737	036240						
	022252	000404			JSR	PC,PUT			:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	022254	000240			BR	10\$:GO TO 10\$ IF NO ERROR
	022256	104000			NOP				:RETURN HERE IF ERROR
	022260	000137	022430		EMT				:ERROR # DEFINED BY PUT SUBROUTINE
1286	022264				JMP	16\$:GO TO 16\$ IF ERROR
1287							10\$:		
1288									:WAIT FOR READ TO COMPLETE AND GET STATUS
1289	022264	004737	036602		JSR	PC,TIMOUT			:WAIT FOR READ TO COMPLETE
1290									
1291	022270	004737	035770						
	022274	000404			JSR	PC,GET			:GO READ REGISTER(S) WITH GET SUBROUTINE
	022276	000240			BR	11\$:GO TO 11\$ IF NO ERROR
	022300	104000			NOP				:RETURN HERE IF ERROR
	022302	000137	022430		EMT				:ERROR # DEFINED BY GET SUBROUTINE
1292	022306				JMP	16\$:GO TO 16\$ IF ERROR
1293							11\$:		
1294									:VERIFY THE RESULTS OF READ OPERATION
1295	022306	004737	036766						
	022312	000405			JSR	PC,PRIERR			:GO CHECK FOR PRIMARY ERRORS
	022314	000240			BR	12\$:GO TO 12\$ IF NO ERROR
	022316	104000			NOP				:RETURN HERE IF ERROR
	022320	004736			EMT				:ERROR # DEFINED BY PRIERR SUBROUTINE
	022322	000137	022430		JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
1296	022326				JMP	16\$:GO TO 16\$ IF ERROR

```

1297 022326 004737 051502      JSR    PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      022332 000405          BR      13$          ;GO TO 13$ IF NO ERROR
      022334 000240          NOP          ;RETURN HERE IF ERROR
      022336 104000          EMT          ;ERROR # DEFINED BY DTASTS SUBROUTINE
      022340 004736          JSR    PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      022342 000137 022430      JMP      16$          ;GO TO 16$ IF ERROR
1298 022346          ;
1299 022346 004737 037620      JSR    PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
      022352 000405          BR      14$          ;GO TO 14$ IF NO ERROR
      022354 000240          NOP          ;RETURN HERE IF ERROR
      022356 104000          EMT          ;ERROR # DEFINED BY SECERR SUBROUTINE
      022360 004736          JSR    PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      022362 000137 022430      JMP      16$          ;GO TO 16$ IF ERROR
1300 022366          ;
1301          ;
1302          ;
1303 022366 004737 035342      ;VERIFY DATA
      022372 101360          JSR    PC,CMPBUF     ;GO COMPARE WRITE, READ DATA BUFFERS
      022374 102364          .WORD    BUFO1E     ;STARTING ADDRESS OF WRITE BUFFER
      022376 000404          .WORD    BUFTWO     ;STARTING ADDRESS OF READ BUFFER
      022400 000240          BR      15$          ;GO TO 15$ IF NO ERROR
      022402 104000          NOP          ;RETURN HERE IF ERROR
      022404 000137 022430      EMT          ;ERROR # DEFINED BY CMPBUF SUBROUTINE
      JMP      16$          ;GO TO 16$ IF ERROR
1304          ;
1305          ;
1306 022410          ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
1307 022410 105237 001421      15$:
1308 022414 123737 001421 001335 INCB    RMDAO+1      ;ADVANCE TRACK COUNT
      CMPB   RMDAO+1,LSTRK+1 ;LAST TRACK ?
1309 022422 101002          BHI     16$          ;YES!
1310 022424 000137 021736      JMP      1$          ;GO DO NEXT SECTOR
1311 022430          ;
1312          ;
1313          ;
      ;*****
      ;*TEST 17      FORMAT PRIME CYLINDERS
      ;*****
      TST17:
1314 022430          SCOPE          ;SCOPE CALL
1315 022430 000004          NOP          ;START OF TEST
      022432 000240          MOV     #STACK,SP   ;INITIALIZE STACK POINTER
      022434 012706 001100      MOV     $BASE,RO    ;RO - UNIBUS ADDRESS
      022440 013700 001276      MOV     TSTQUE,R1   ;(R1) = DEVICE BEING TESTED
      022444 013701 001466      MOV     #17,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
      022450 012737 000017 001226
1316 022456 012737 000001 001446 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
1317 022464 012737 000000 001420      MOV     #1,RMDCO    ;CYLINDER = 1
      1$:
1318 022472          MOV     #0,RMDAO     ;TRACK = 0, SECTOR - 0
1319 022472 012737 010000 001444      MOV     #FMT16,RMOFO ;16 BIT FORMAT
1320 022500 012737 177376 001414      MOV     #-258,RMWCO ;2 + 256 WORDS (2'S COMP)
1321 022506 012737 101360 001416      MOV     #BUFO1E,RMBAO ;DATA BUFFER ADDRESS
1322 022514 012737 000062 001412      MOV     #WH,RMCS10  ;WRITE HEADER AND DATA
1323          ;
1324          ;
      ;VERIFY THAT SECTOR IS NOT BAD
1325 022522 004737 033152      JSR    PC,BADSCT    ;CALL BAD SECTOR MODULE
      022526 000405          BR      2$          ;GO TO 2$ IF NO ERROR
      022530 104401 063346      TYPE    ,SCTMSG     ;TYPE BAD SECTOR MESSAGE
      022534 104000          EMT          ;ERROR # DEFINED BY BADSCT SUBROUTINE
    
```

```

1325 022536 000137 023172          JMP      16$          :GO TO 16$ IF ERROR
1326 022542          2$:
1326 022542 012737 001446 001174   MOV      #RMDLO,$TMPO :USE CYLINDER FOR DATA PATTERN
1327 022550 012737 000001 001176   MOV      #1,$TMP1
1328 022556 004737 035104          JSR      PC,GENBUF   :GO GENERATE DATA BUFFER
1329
1330 022562          3$:
1331
1332          ;PREPARE DEVICE FOR DATA TRANSFER
1333 022562 004737 032176   JSR      PC,TSTPRP   :PREPARE DEVICE FOR TEST
      022566 154130          .WORD    154130     :TASK DESCRIPTOR AS FOLLOWS:
      :SELECT DEVICE & VERIFY DEVICE AVAILABLE
      :CLEAR CONTROLLER & SELECT DEVICE
      :VERIFY CONTROLLER CLEAR OPERATION
      :PACK ACKNOWLEDGE IF VOLUME NOT VALID
      :VERIFY PACK ACKNOWLEDGE
      :RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      :VERIFY RECALIBRATION
      :GO TO 4$ IF NO ERROR
      :RETURN HERE IF ERROR
      :ERROR # DEFINED BY TSTPRP SUBROUTINE
      :GO TO 16$ IF ERROR
      022570 000404          BR       4$
      022572 000240          NOP
      022574 104000          EMT
      022576 000137 023172   JMP      16$
1334 022602          4$:
1335
1336          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1337 022602 012737 000063 001412   MOV      #WH!GO,RMCS1G :WRITE HEADER AND DATA
1338 022610 012702 001553          MOV      #PUTINX,R2    :WRITE REGISTER INDEX TABLE
1339 022614 112722 000006          MOVB     #RMDA,(R2)+
1340 022620 112722 000034          MOVB     #RMDC,(R2)+
1341 022624 112722 000032          MOVB     #RMOF,(R2)+
1342 022630 112722 000002          MOVB     #RMWC,(R2)+
1343 022634 112722 000004          MOVB     #RMBA,(R2)+
1344 022640 112722 000000          MOVB     #RMCS1,(R2)+
1345 022644 112722 000200          MOVB     #200,(R2)+
1346 022650 004737 036240   JSR      PC,PUT       :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      022654 000404          BR       5$          :GO TO 5$ IF NO ERROR
      022656 000240          NOP                :RETURN HERE IF ERROR
      022660 104000          EMT                :ERROR # DEFINED BY PUT SUBROUTINE
      022662 000137 023172   JMP      16$          :GO TO 16$ IF ERROR
1347 022666          5$:
1348
1349          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1350 022666 004737 035704   JSR      PC,GETSTS
1351
1352          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
1353 022672 004737 036602   JSR      PC,TIMOUT    :WAIT FOR COMMAND TO COMPLETE
1354 022676 004737 035770   JSR      PC,GET       :GO READ REGISTER(S) WITH GET SUBROUTINE
      022702 000404          BR       6$          :GO TO 6$ IF NO ERROR
      022704 000240          NOP                :RETURN HERE IF ERROR
      022706 104000          EMT                :ERROR # DEFINED BY GET SUBROUTINE
      022710 000137 023172   JMP      16$          :GO TO 16$ IF ERROR
1355 022714          6$:
1356
1357          ;VERIFY RESULTS OF WRITE COMMAND
1358 022714 004737 036766   JSR      PC,PRIERR    :GO CHECK FOR PRIMARY ERRORS
      022720 000405          BR       7$          :GO TO 7$ IF NO ERROR
      022722 000240          NOP                :RETURN HERE IF ERROR
  
```

	022724	104000			EMT				:ERROR # DEFINED BY PRIERR SUBROUTINE
	022726	004736			JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
	022730	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1359	022734						7\$:		
1360	022734	004737	051502		JSR	PC,DTASTS			:GO VERIFY RESULTS OF DATA TRANSFER
	022740	000405			BR	8\$:GO TO 8\$ IF NO ERROR
	022742	000240			NOP				:RETURN HERE IF ERROR
	022744	104000			EMT				:ERROR # DEFINED BY DTASTS SUBROUTINE
	022746	004736			JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
	022750	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1361	022754						8\$:		
1362	022754	004737	037620		JSR	PC,SECERR			:GO CHECK FOR SECONDARY ERRORS
	022760	000405			BR	9\$:GO TO 9\$ IF NO ERROR
	022762	000240			NOP				:RETURN HERE IF ERROR
	022764	104000			EMT				:ERROR # DEFINED BY SECERR SUBROUTINE
	022766	004736			JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
	022770	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1363	022774						9\$:		
1364									
1365									
1366	022774	012737	000073	001412	MOV	#RH.GO,RMCS10			:READ HEADER & DATA COMMAND
1367	023002	012737	102364	001416	MOV	#BUFTWO,RMBA0			:CHANGE BUS ADDRESS
1368	023010	004737	036240		JSR	PC,PUT			:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	023014	000404			BR	10\$:GO TO 10\$ IF NO ERROR
	023016	000240			NOP				:RETURN HERE IF ERROR
	023020	104000			EMT				:ERROR # DEFINED BY FJT SUBROUTINE
	023022	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1369	023026						10\$:		
1370									
1371									
1372	023026	004737	036602		JSR	PC,TIMOUT			:WAIT FOR READ TO COMPLETE
1373	023032	004737	035770		JSR	PC,GET			:GO READ REGISTER(S) WITH GET SUBROUTINE
	023036	000404			BR	11\$:GO TO 11\$ IF NO ERROR
	023040	000240			NOP				:RETURN HERE IF ERROR
	023042	104000			EMT				:ERROR # DEFINED BY GET SUBROUTINE
	023044	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1374	023050						11\$:		
1375									
1376									
1377	023050	004737	036766		JSR	PC,PRIERR			:GO CHECK FOR PRIMARY ERRORS
	023054	000405			BR	12\$:GO TO 12\$ IF NO ERROR
	023056	000240			NOP				:RETURN HERE IF ERROR
	023060	104000			EMT				:ERROR # DEFINED BY PRIERR SUBROUTINE
	023062	004736			JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
	023064	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1378	023070						12\$:		
1379	023070	004737	051502		JSR	PC,DTASTS			:GO VERIFY RESULTS OF DATA TRANSFER
	023074	000405			BR	13\$:GO TO 13\$ IF NO ERROR
	023076	000240			NOP				:RETURN HERE IF ERROR
	023100	104000			EMT				:ERROR # DEFINED BY DTASTS SUBROUTINE
	023102	004736			JSR	PC,@(SP)+			:GO BACK FOR MORE ERROR CHECKS
	023104	000137	023172		JMP	16\$:GO TO 16\$ IF ERROR
1380	023110						13\$:		
1381	023110	004737	037620		JSR	PC,SECERR			:GO CHECK FOR SECONDARY ERRORS
	023114	000405			BR	14\$:GO TO 14\$ IF NO ERROR
	023116	000240			NOP				:RETURN HERE IF ERROR
	023120	104000			EMT				:ERROR # DEFINED BY SECERR SUBROUTINE

```
1382 023122 004736 JSR PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
1383 023124 000137 023172 JMP 16$ ;GO TO 16$ IF ERROR
1384 023130
1385 023130 004737 035342 ;VERIFY DATA
023134 101360 JSR PC,CMPBUF ;GO COMPARE WRITE, READ DATA BUFFERS
023136 102364 .WORD BUFO1E ;STARTING ADDRESS OF WRITE BUFFER
023140 000404 .WORD BUFTWO ;STARTING ADDRESS OF READ BUFFER
023142 000240 BR 15$ ;GO TO 15$ IF NO ERROR
023144 104000 NOP ;RETURN HERE IF ERROR
023146 000137 023172 EMT ;ERROR # DEFINED BY CMPBUF SUBROUTINE
1386 023152 JMP 16$ ;GO TO 16$ IF ERROR
1387
1388 ;INCREMENT ADDRESS AND FORMAT NEXT PRIME CYLINDER
1389 023152 006337 001446 ASL RMDCO ;ADVANCE CYLINDER COUNT
1390 023156 023727 001446 001000 CMP RMDCO,#512. ;DONE ALL PRIME CYLINDERS ?
1391 023164 003002 BGT 16$ ;YES!!
1392 023166 000137 022472 JMP 1$ ;GO DO NEXT CYLINDER
1393 023172
1394
1395 ;*****
;*TEST 20 READ HEADER & DATA IN LAST SECTOR
;*****
TST20:
023172 000004 SCOPE ;SCOPE CALL
023174 000240 NOP ;START OF TEST
023176 012706 001100 MOV #STACK,SP ;INITIALIZE STACK POINTER
023202 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
023206 013701 001466 MOV TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
023212 012757 000020 001226 MOV #20,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
1396
1397 ;SETUP PARAMETERS FOR READING LAST SECTOR
1398 023220 012737 001466 001446 MOV #822.,RMDCO ;LAST CYLINDER
1399 023226 013737 001334 001420 MOV LSTRK,RMDAO ;SET LAST TRACK AND
1400 023234 112737 000036 001420 MOV# #30.,RMDAO ;LAST SECTOR
1401 023242 012737 010000 001444 MOV #FMT16,RMFO ;16 BIT FORMAT
1402 023250
1403 023250 012737 177376 001414 MOV #-258.,RMWCO ;2 + 256 WORDS (2'S COMP)
1404 023256 012737 102364 001416 MOV #BUFTWO,RMBAO ;DATA BUFFER ADDRESS
1405 023264 012737 000073 001412 MOV #RH!GO,RMCS10 ;READ HEADER AND DATA
1406
1407 ;PREPARE DEVICE FOR DATA TRANSFER
1408 023272 004737 032176 JSR PC,TSTPRP ;PREPARE DEVICE FOR TEST
023276 154130 .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
;SELECT DEVICE & VERIFY DEVICE AVAILABLE
;CLEAR CONTROLLER & SELECT DEVICE
;VERIFY CONTROLLER CLEAR OPERATION
;PACK ACKNOWLEDGE IF VOLUME NOT VALID
;VERIFY PACK ACKNOWLEDGE
;RECALIBRATE IF "SKI" OR "PIP" IS SET
;VERIFY RECALIBRATION
023300 000404 BR 2$ ;GO TO 2$ IF NO ERROR
023302 000240 NOP ;RETURN HERE IF ERROR
023304 104000 EMT ;ERROR # DEFINED BY TSTPRP SUBROUTINE
023306 000137 023476 . JMP 9$ ;GO TO 9$ IF ERROR
1409 023312
2$:
```

```

1410 023312 012702 001553      MOV      #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
1411 023315 112722 C00006      MOVB     #RMDA,(R2)+
1412 023322 112722 000034      MOVB     #PMDC,(R2)+
1413 023326 112722 000032      MOVB     #RMOF,(R2)+
1414 023332 112722 000004      MOVB     #RMBA,(R2)+
1415 023336 112722 C00002      MOVB     #RMWC,(R2)+
1416 023342 112722 000000      MOVB     #RMCS1,(R2)+
1417 023346 112722 000200      MOVB     #200,(R2)+      ;TERMINATOR
1418 023352
1419
1420
1421 023352 004737 036240      ;READ HEADER AND DATA OF LAST SECTOR
      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      BR      4$          ;GO TO 4$ IF NO ERROR
      NOP
      EMT          ;RETURN HERE IF ERROR
      JMP      9$          ;ERROR # DEFINED BY PUT SUBROUTINE
      ;GO TO 9$ IF ERROR
1422 023370
1423
1424
1425 023370 004737 035704      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
      JSR      PC,GETSTS
1426 023374 004737 036602      JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
1427
1428 023400 004737 035770      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      BR      5$          ;GO TO 5$ IF NO ERROR
      NOP
      EMT          ;RETURN HERE IF ERROR
      JMP      9$          ;ERROR # DEFINED BY GET SUBROUTINE
      ;GO TO 9$ IF ERROR
1429 023416
1430
1431
1432 023416 004737 036766      ;VERIFY THE RESULTS OF READ OPERATION
      JSR      PC,PRIERR   ;GO CHECK FOR PRIMARY ERRORS
      BR      6$          ;GO TO 6$ IF NO ERROR
      NOP
      EMT          ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR      PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      JMP      9$          ;GO TO 9$ IF ERROR
1433 023436
1434 023436 004737 051502      6$:      JSR      PC,DTASTS     ;GO VERIFY RESULTS OF DATA TRANSFER
      BR      7$          ;GO TO 7$ IF NO ERROR
      NOP
      EMT          ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR      PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      JMP      9$          ;GO TO 9$ IF ERROR
1435 023456
1436 023456 004737 037620      7$:      JSR      PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
      BR      8$          ;GO TO 8$ IF NO ERROR
      NOP
      EMT          ;RETURN HERE IF ERROR
      ;ERROR # DEFINED BY SECERR SUBROUTINE
      JSR      PC,@(SP)+   ;GO BACK FOR MORE ERROR CHECKS
      JMP      9$          ;GO TO 9$ IF ERROR
1437 023476
1438
1439 023476
1440
1441

```

```

:*****
:*TEST 21      READ HEADER & DATA W/ AOE ERROR
:*****

```

```

023476
023476 000004
023500 000240
023502 012706 001100
023506 013700 001276
023512 013701 001466
023516 012737 000021 001226
1442
1443
1444 023524 013737 001334 001420
1445 023532 112737 000036 001420
1446 023540 012737 010000 001444
1447 023546
1448 023546 012737 001466 001446
1449 023554 012737 176774 001414
1450 023562 012737 102364 001416
1451 023570 012737 000073 001412
1452
1453
1454 023576 004737 032176
    023602 154130

023604 000404
023606 000240
023610 104000
023612 000137 024002
1455 023616
1456 023616 012702 001553
1457 023622 112722 000006
1458 023626 112722 000034
1459 023632 112722 000032
1460 023636 112722 000002
1461 023642 112722 000004
1462 023646 112722 000000
1463 023652 112722 000200
1464
1465 023656 004737 036240
    023662 000404
    023664 000240
    023666 104000
    023670 000137 024002
1466 023674
1467
1468
1469 023674 004737 035704
1470 023700 004737 036602
1471
1472 023704 004737 035770
    023710 000404
    023712 000240
    023714 104000

TST21:
SCOPE
NOP
MOV #STACK,SP
MOV $BASE,R0
MOV TSTQUE,R1
MOV #21,$TESTN

:SCOPE CALL
:START OF TEST
:INITIALIZE STACK POINTER
:R0 = UNIBUS ADDRESS
:(R1) = DEVICE BEING TESTED
::SET TEST NUMBER IN APT MAIL BOX

:SETUP PARAMETERS FOR READING 2 SECTORS STARTING WITH LAST SECTOR
MOV LSTRK,RMDAO
MOV #30,RMDAO
MOV #FMT16,RMOFO

:SET LAST TRACK AND
:LAST SECTOR
:16 BIT FORMAT

1$:
MOV #822,RMDCO
MOV #-258,*2,RMWCO
MOV #BUF TWO,RMBAO
MOV #RH!GO,RMCS10

:LAST CYLINDER
:READ 2 SECTORS
:DATA BUFFER ADDRESS
:READ HEADER AND DATA

:PREPARE DEVICE FOR DATA TRANSFER
JSR PC,TSTPRP
.WORD 154130

:PREPARE DEVICE FOR TEST
:TASK DESCRIPTOR AS FOLLOWS:
:SELECT DEVICE & VERIFY DEVICE AVAILABLE
:CLEAR CONTROLLER & SELECT DEVICE
:VERIFY CONTROLLER CLEAR OPERATION
:PACK ACKNOWLEDGE IF VOLUME NOT VALID
:VERIFY PACK ACKNOWLEDGE
:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
:VERIFY RECALIBRATION
:GO TO 2$ IF NO ERROR
:RETURN HERE IF ERROR
:ERROR # DEFINED BY TSTPRP SUBROUTINE
:GO TO 8$ IF ERROR

2$:
MOV #PUTINX,R2
MOV #RMDA,(R2)+
MOV #RMDC,(R2)+
MOV #RMOF,(R2)+
MOV #RMWC,(R2)+
MOV #RMBA,(R2)+
MOV #RMCS1,(R2)+
MOV #200,(R2)+

:WRITE REGISTER INDEX TABLE

JSR PC,PUT
BR 3$
NOP
EMT
JMP 8$

:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
:GO TO 3$ IF NO ERROR
:RETURN HERE IF ERROR
:ERROR # DEFINED BY PUT SUBROUTINE
:GO TO 8$ IF ERROR

3$:

:SETUP INPUT REGISTER BUFFER FOR READING STATUS
JSR PC,GETSTS
JSR PC,TIMOUT

:WAIT FOR READ TO COMPLETE

JSR PC,GET
BR 4$
NOP
EMT

:GO READ REGISTER(S) WITH GET SUBROUTINE
:GO TO 4$ IF NO ERROR
:RETURN HERE IF ERROR
:ERROR # DEFINED BY GET SUBROUTINE
    
```



```

1473 023716 000137 024002          JMP      8$          :GO TO 8$ IF ERROR
1474 023722
1475
1476 023722 004737 036766          :VERIFY THE RESULTS OF READ OPERATION
      023726 000405          JSR      PC,PRIERR  :GO CHECK FOR PRIMARY ERRORS
      023730 000240          BR       5$          :GO TO 5$ IF NO ERROR
      023732 104000          NOP      :RETURN HERE IF ERROR
      023734 004736          EMT     :ERROR # DEFINED BY PRIERR SUBROUTINE
      023736 000137 024002          JSR      PC,@(SP)+  :GO BACK FOR MORE ERROR CHECKS
      023736 000137 024002          JMP      8$          :GO TO 2$ IF ERROR
1477 023742
1478 023742 004737 051502          5$:
      023746 000405          JSR      PC,DTASTS  :GO VERIFY RESULTS OF DATA TRANSFER
      023750 000240          BR       6$          :GO TO 6$ IF NO ERROR
      023752 104000          NOP      :RETURN HERE IF ERROR
      023754 004736          EMT     :ERROR # DEFINED BY DTASTS SUBROUTINE
      023756 000137 024002          JSR      PC,@(SP)+  :GO BACK FOR MORE ERROR CHECKS
      023756 000137 024002          JMP      8$          :GO TO 8$ IF ERROR
1479 023762
1480 023762 004737 037620          6$:
      023766 000405          JSR      PC,SECERR  :GO CHECK FOR SECONDARY ERRORS
      023770 000240          BR       7$          :GO TO 7$ IF NO ERROR
      023772 104000          NOP      :RETURN HERE IF ERROR
      023774 004736          EMT     :ERROR # DEFINED BY SECERR SUBROUTINE
      023776 000137 024002          JSR      PC,@(SP)+  :GO BACK FOR MORE ERROR CHECKS
      023776 000137 024002          JMP      8$          :GO TO 8$ IF ERROR
1481 024002
1482
1483 024002
1484
1485

```

```

*****
: *TEST 22          READ INVALID SECTOR ADDRESS
*****
TST22:

```

```

024002
024002 000004          :SCOPE CALL
024004 000240          :START OF TEST
024006 012706 001100          MOV      #STACK,SP  :INITIALIZE STACK POINTER
024012 013700 001276          MOV      $BASE,R0   :R0 = UNIBUS ADDRESS
024016 013701 001466          MOV      TSTQUE,R1  :(R1) = DEVICE BEING TESTED
024022 012737 000022 001226          MOV      #22,$TESTN :;SET TEST NUMBER IN APT MAIL BOX
1486
1487 024030 012737 000000 001446          MOV      #0,RMDCO   :CYLINDER = 0
1488 024036 012737 000036 001420          MOV      #30,RMDAO  :TRACK = 0, INVALID SECTOR - 30.
1489 024044 012737 000000 001444          MOV      #0,RMOFO   :18 BIT FORMAT
1490 024052 012737 101360 001416          MOV      #BUFONE,RMBAO :CHANGE BUS ADDRESS
1491 024060 012737 177376 001414          MOV      #-258,RMWCO :2 + 256 WORDS (2'S COMP)
1492 024066 012737 000073 001412          MOV      #RH.GO,RMCSTO :READ HEADER AND DATA
1493 024074
1494
1495
1496 024074 004737 032176          1$:
      024100 154130          :PREPARE DEVICE FOR DATA TRANSFER
      024100 154130          JSR      PC,TSTPPP  :PREPARE DEVICE FOR TEST
      024100 154130          .WORD   154130     :TASK DESCRIPTOR AS FOLLOWS:
      024100 154130          :SELECT DEVICE & VERIFY DEVICE AVAILABLE
      024100 154130          :CLEAR CONTROLLER & SELECT DEVICE
      024100 154130          :VERIFY CONTROLLER CLEAR OPERATION
      024100 154130          :PACK ACKNOWLEDGE IF VOLUME NOT VALID
      024100 154130          :VERIFY PACK ACKNOWLEDGE
      024100 154130          :RECALIBRATE IF 'SKI' OR 'PIP' IS SET
      024100 154130          :VERIFY RECALIBRATION

```

```

024102 000404 BR 2$ :GO TO 2$ IF NO ERROR
024104 000240 NOP :RETURN HERE IF ERROR
024106 104000 EMT :ERROR # DEFINED BY TSTPRP SUBROUTINE
024110 000137 024344 JMP 9$ :GO TO 9$ IF ERROR
1497 024114 2$:
1498
1499 :SETUP AND EXECUTE READ HEADER AND DATA COMMAND
1500 024114 012702 001553 MOV #PUTINX,R2 :LOAD REGISTER INDEX TABLE
1501 024120 112722 000006 MOVB #RMDA,(R2)+
1502 024124 112722 000034 MOVB #RMDC,(R2)+
1503 024130 112722 000032 MOVB #RMOF,(R2)+
1504 024134 112722 000002 MOVB #RMWC,(R2)+
1505 024140 112722 000004 MOVB #RMAA,(R2)+
1506 024144 112722 000000 MOVB #RMCS1,(R2)+
1507 024150 112722 000200 MOVB #200,(R2)+ :SET TERMINATOR BYTE
1508
1509 024154 004737 036240 JSR PC,PUT :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
024160 000404 BR 3$ :GO TO 3$ IF NO ERROR
024162 000240 NOP :RETURN HERE IF ERROR
024164 104000 EMT :ERROR # DEFINED BY PUT SUBROUTINE
024166 000137 024344 JMP 9$ :GO TO 9$ IF ERROR
1510 024172 3$:
1511
1512 :SETUP INPUT REGISTER BUFFER FOR READING STATUS
1513 024172 004737 035704 JSR PC,GETSTS
1514 024176 004737 036602 JSR PC,TIMOUT :WAIT FOR COMMAND TO COMPLETE
1515
1516 024202 004737 035770 JSR PC,GET :GO READ REGISTER(S) WITH GET SUBROUTINE
024206 000404 BR 4$ :GO TO 4$ IF NO ERROR
024210 000240 NOP :RETURN HERE IF ERROR
024212 104000 EMT :ERROR # DEFINED BY GET SUBROUTINE
024214 000137 024344 JMP 9$ :GO TO 9$ IF ERROR
1517 024220 4$:
1518
1519 :VERIFY RESULTS OF READ COMMAND
1520 024222 004737 036766 JSR PC,PRIERR :GO CHECK FOR PRIMARY ERRORS
024224 000405 BR 5$ :GO TO 5$ IF NO ERROR
024226 000240 NOP :RETURN HERE IF ERROR
024230 104000 EMT :ERROR # DEFINED BY PRIERR SUBROUTINE
024232 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
024234 000137 024344 JMP 9$ :GO TO 9$ IF ERROR
1521 024240 5$:
1522 024240 004737 051502 JSR PC,DTASTS :GO VERIFY RESULTS OF DATA TRANSFER
024244 000405 BR 6$ :GO TO 6$ IF NO ERROR
024246 000240 NOP :RETURN HERE IF ERROR
024250 104000 EMT :ERROR # DEFINED BY DTASTS SUBROUTINE
024252 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
024254 000137 024344 JMP 9$ :GO TO 9$ IF ERROR
1523 024260 6$:
1524 024260 004737 037620 JSR PC,SECERR :GO CHECK FOR SECONDARY ERRORS
024264 000405 BR 7$ :GO TO 7$ IF NO ERROR
024266 000240 NOP :RETURN HERE IF ERROR
024270 104000 EMT :ERROR # DEFINED BY SECERR SUBROUTINE
024272 004736 JSR PC,@(SP)+ :GO BACK FOR MORE ERROR CHECKS
024274 000137 024344 JMP 9$ :GO TO 9$ IF ERROR
1525 024300 7$:
1526

```

```

1527          ;INCREMENT ADDRESS AND PEAD NEXT SECTOR
1528 024300 005237 001420      INC      RMDAO      ;ADVANCE SECTOR COUNT
1529 024304 023727 001420 000100  CMP      RMDAO,#64.  ;DONE ALL SECTORS??
1530 024312 101412          BLOS     8$         ;NO !!
1531
1532 024314 032737 010000 001444  BIT      #FMT16,RMOFO ;TEST 16 BIT MODE YET ?
1533 024322 001010          BNE      9$         ;YES !!
1534 024324 012737 000040 001420  MOV      #32.,RMDAO  ;TRACK = 0, INVAID SECTOR = 32.
1535 024332 012737 010000 001444  MOV      #FMT16,RMOFO ;SET 16 BIT MODE IN OFFSET AND
1536 024340 000137 024074 8$:   JMP      1$         ;TEST AGAIN.
1537
1538 024344          9$:
1539
1540          ;*****
          ;*TEST 23      READ INVALID TRACK ADDRESS
          ;*****
          TST23:
          SCOPE          ;SCOPE CALL
          NOP            ;START OF TEST
          MOV      #STACK,SP ;INITIALIZE STACK POINTER
          MOV      $BASE,R0  ;R0 = UNIBUS ADDRESS
          MOV      TSTQUE,R1 ; (R1) - DEVICE BEING TESTED
          MOV      #23,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
1541
1542 024372 012737 000000 001446  MOV      #0,RMDCO    ;CYLINDER = 0
1543 024400 013737 001334 001420  MOV      LSTRK,RMDAO ;LOAD LAST TRACK, SECTOR - 0
1544 024406 105237 001421          INCB     RMDAO+1    ;INCREMENT TO FIRST INVALID TRACK
1545 024412 012737 010000 001444  MOV      #FMT16,RMOFO ;16 BIT FORMAT
1546 024420 012737 177376 001414  MOV      #-258.,RMWCO ;2 + 256 WORDS (2'S COMP)
1547 024426 012737 101360 001416  MOV      #BUFONE,RMBAO ;DATA BUFFER ADDRESS
1548 024434 012737 000073 001412  MOV      #RH.GO,RMCSI0 ;READ HEADER AND DATA
1549 024442
1550
1551          ;PREPARE DEVICE FOR DATA TRANSFER
1552 024442 004737 032176  JSR      PC,TSTPRP  ;PREPARE DEVICE FOR TEST
          .WORD     154130 ;TASK DESCRIPTOR AS FOLLOWS:
          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
          ;CLEAR CONTROLLER & SELECT DEVICE
          ;VERIFY CONTROLLER CLEAR OPERATION
          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
          ;VERIFY PACK ACKNOWLEDGE
          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
          ;VERIFY RECALIBRATION
          BR      2$     ;GO TO 2$ IF NO ERROR
          NOP          ;RETURN HERE IF ERROR
          EMT          ;ERROR # DEFINED BY TSTPRP SUBROUTINE
          JMP      8$     ;GO TO 8$ IF ERROR
1553 024462 000404          BR      2$
1554 024452 000240          NOP
1555 024454 104000          EMT
1556 024456 000137 024666  JMP      8$
1557
1558          ;SETUP AND EXECUTE READ HEADER AND DATA COMMAND
1559 024462 012702 001553  MOV      #PUTINX,R2  ;LOAD REGISTER INDEX TABLE
1560 024466 112722 000006  MOVB     #RMDA,(R2)+
1561 024472 112722 000034  MOVB     #RMDC,(R2)+
1562 024476 112722 000032  MOVB     #RMOF,(R2)+
1563 024502 112722 000002  MOVB     #RMWC,(R2)+
1564 024506 112722 000004  MOVB     #RMBA,(R2)+
1565 024512 112722 000000  MOVB     #RMCS1,(R2)+
    
```

```

1563 024516 112722 000200      MOVB    #200,(R2)      ;SET TERMINATOR BYTE
1564
1565 024522 004737 036240      JSR     PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      024526 000404      BR      3$           ;GO TO 3$ IF NO ERROR
      024530 000240      NOP                    ;RETURN HERE IF ERROR
      024532 104000      EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
      024534 000137 024666      JMP     8$           ;GO TO 8$ IF ERROR
1566 024540      3$:
1567
1568      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1569 024540 004737 035704      JSR     PC,GETSTS
1570 024544 004737 036602      JSR     PC,TIMOUT    ;WAIT FOR COMMAND TO COMPLETE
1571
1572 024550 004737 035770      JSR     PC,GET        ;GO READ REGISTER(S) WITH GET SUBROUTINE
      024554 000404      BR      4$           ;GO TO 4$ IF NO ERROR
      024556 000240      NOP                    ;RETURN HERE IF ERROR
      024560 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
      024562 000137 024666      JMP     8$           ;GO TO 8$ IF ERROR
1573 024566      4$:
1574
1575      ;VERIFY RESULTS OF READ COMMAND
1576 024566 004737 036766      JSR     PC,PRIERR    ;GO CHECK FOR PRIMARY ERRORS
      024572 000405      BR      5$           ;GO TO 5$ IF NO ERROR
      024574 000240      NOP                    ;RETURN HERE IF ERROR
      024576 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      024600 004736      JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      024602 000137 024666      JMP     8$           ;GO TO 8$ IF ERROR
1577 024606      5$:
1578 024606 004737 051502      JSR     PC,DTASTS    ;GO VERIFY RESULTS OF DATA TRANSFER
      024612 000405      BR      6$           ;GO TO 6$ IF NO ERROR
      024614 000240      NOP                    ;RETURN HERE IF ERROR
      024616 104000      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
      024620 004736      JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      024622 000137 024666      JMP     8$           ;GO TO 8$ IF ERROR
1579 024626      6$:
1580 024626 004737 037620      JSR     PC,SECERR    ;GO CHECK FOR SECONDARY ERRORS
      024632 000405      BR      7$           ;GO TO 7$ IF NO ERROR
      024634 000240      NOP                    ;RETURN HERE IF ERROR
      024636 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      024640 004736      JSR     PC,@(SP)+    ;GO BACK FOR MORE ERROR CHECKS
      024642 000137 024666      JMP     8$           ;GO TO 8$ IF ERROR
1581 024646      7$:
1582
1583      ;INCREMENT ADDRESS AND READ NEXT TRACK
1584 024646 105237 001421      INCB   RMDAO+1      ;ADVANCE TRACK COUNT
1585 024652 123727 001421 000200  CMPB   RMDAO+*,#128. ;DONE ALL TRACKS??
1586 024660 101002      BHI    8$           ;YES!!
1587 024662 000137 024442      JMP     1$           ;GO DO NEXT TRACK
1588 024666      8$:
1589
1590
*****
:TEST 24      READ INVALID CYLINDER ADDRESS
*****
TST24:
024666      SCOPE      ;SCOPE CALL
024666 000004      NOP      ;START OF TEST
024670 000240      MOV     #STACK,SP ;INITIALIZE STACK POINTER
024672 012706 001100

```

```
024676 013700 001276          MOV    $BASE,R0      ;RO = UNIBUS ADDRESS
024702 013701 001466          MOV    TSTQUE,R1     ;(R1) = DEVICE BEING TESTED
024706 012737 000024 001226    MOV    #24,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX

1591
1592 024714 012737 001467 001446    MOV    #823.,RMDCO   ;START AT FIRST INVALID CYLINDER
1593 024722 012737 000000 001420    MOV    #0,RMDAO      ;TRACK = 0, SECTOR = 0
1594 024730 012737 010000 001444    MOV    #FMT16,RMOFO  ;16 BIT FORMAT
1595 024736 012737 177376 001414    MOV    #-258.,RMMCO  ;2 + 256 WORDS (2'S COMP)
1596 024744 012737 101360 001416    MOV    #BUFONE,RMBAO ;DATA BUFFER ADDRESS
1597 024752 012737 000073 001412    MOV    #R1.GO,RMCS10 ;READ HEADER AND DATA
1598 024760
1599
1600
1601 024760 004737 032176          ;PREPARE DEVICE FOR DATA TRANSFER
      024764 154130          JSR    PC,TSTPRP     ;PREPARE DEVICE FOR TEST
                          .WORD 154130 ;TASK DESCRIPTOR AS FOLLOWS:
                          ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                          ;CLEAR CONTROLLER & SELECT DEVICE
                          ;VERIFY CONTROLLER CLEAR OPERATION
                          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                          ;VERIFY PACK ACKNOWLEDGE
                          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                          ;VERIFY RECALIBRATION
                          ;GO TO 2$ IF NO ERROR
024766 000404          BR     2$           ;RETURN HERE IF ERROR
024770 000240          NOP
024772 104000          EMT
024774 000137 025204          JMP    8$           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                          ;GO TO 8$ IF ERROR

1602 025000          2$:
1603
1604
1605 025000 012702 001553          ;SETUP AND EXECUTE READ HEADER AND DATA COMMAND
      025004 112722 000006    MOV    #PUTINX,R2   ;LOAD REGISTER INDEX TABLE
1607 025010 112722 000034    MOVB  #RMDA,(R2)+
1608 025014 112722 000032    MOVB  #RMDC,(R2)+
1609 025020 112722 000002    MOVB  #RMOF,(R2)+
1610 025024 112722 000004    MOVB  #RMMC,(R2)+
1611 025030 112722 000000    MOVB  #RMBA,(R2)+
1612 025034 112722 000200    MOVB  #RMCS1,(R2)+
1613
1614 025040 004737 036240          ;SET TERMINATOR BYTE
      025044 000404          JSR    PC,PUT       ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      025046 000240          BR     3$           ;GO TO 3$ IF NO ERROR
      025050 104000          NOP                ;RETURN HERE IF ERROR
      025052 000137 025204          EMT                ;ERROR # DEFINED BY PUT SUBROUTINE
1615 025056          JMP    8$           ;GO TO 8$ IF ERROR

1616
1617
1618 025056 004737 035704          3$:
1619 025062 004737 036602          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
      025066 004737 035770          JSR    PC,GETSTTS  ;WAIT FOR COMMAND TO COMPLETE
      025072 000404          JSR    PC,GET
      025074 000240          BR     4$           ;GO READ REGISTER(S) WITH GET SUBROUTINE
      025076 104000          BR     4$           ;GO TO 4$ IF NO ERROR
      025100 000137 025204          NOP                ;RETURN HERE IF ERROR
1622 025104          EMT                ;ERROR # DEFINED BY GET SUBROUTINE
1623          JMP    8$           ;GO TO 8$ IF ERROR
1624
      ;VERIFY RESULTS OF READ COMMAND
```

```

1625 025104 004737 036760      JSR    PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      025110 000405      BR     5$             ;GO TO 5$ IF NO ERROR
      025112 000240      NOP                    ;RETURN HERE IF ERROR
      025114 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      025116 004736      JSR    PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
      025120 000137 025204      JMP    8$             ;GO TO 8$ IF ERROR
1626 025124      5$:
1627 025124 004737 051502      JSR    PC,DTASTS     ;GO VERIFY RESULTS OF DATA TRANSFER
      025130 000405      BR     6$             ;GO TO 6$ IF NO ERROR
      025132 000240      NOP                    ;RETURN HERE IF ERROR
      025134 104000      EMT                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
      025136 004736      JSR    PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
      025140 000137 025204      JMP    8$             ;GO TO 8$ IF ERROR
1628 025144      6$:
1629 025144 004737 037620      JSR    PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
      025150 000405      BR     7$             ;GO TO 7$ IF NO ERROR
      025152 000240      NOP                    ;RETURN HERE IF ERROR
      025154 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      025156 004736      JSR    PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
      025160 000137 025204      JMP    8$             ;GO TO 8$ IF ERROR
1630 025164      7$:
1631
1632      ;INCREMENT ADDRESS AND READ NEXT CYLINDER
1633 025164 005237 001446      INC    RMDCO         ;ADVANCE CYLINDER COUNT
1634 025170 023727 001446 002000  CMP    RMDCO,#1024.  ;DONE ALL CYLINDERS??
1635 025176 002002      BGE    8$             ;YES.!
1636 025200 000137 024760      JMP    1$             ;GO DO NEXT SECTOR
1637 025204      8$:
1638
1639      ;*****
      ;*TEST 25      FORMAT AT OFFSET
      ;*****
      TST25:
      025204      SCOPE                    ;SCOPE CALL
      025204 000004      NOP                    ;START OF TEST
      025206 000240      MOV    #STACK,SP     ;INITIALIZE STACK POINTER
      025210 012706 001100      MOV    $BASE,R0      ;R0 - UNIBUS ADDRESS
      025214 013700 001276      MOV    TSTQUE,R1     ;(R1) = DEVICE BEING TESTED
      025220 013701 001466      MOV    #25,$TESTN    ;SET TEST NUMBER IN APT MAIL BOX
      025224 012737 000025 001226
1640
1641      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
1642 025232 012737 000000 001446  MOV    #0,RMDCO      ;CYLINDER = 0
1643 025240 012737 000000 001420  MOV    #0,RMDAO      ;TRACK 0, SECTOR 0
1644 025246 012737 010000 001444  MOV    #FMT16,RMOFO  ;16 BIT FORMAT
1645 025254 012737 177376 001414  MOV    #-258.,RMWCO  ;2 + 256 WORDS (2'S COMP)
1646 025262 012737 101360 001416  MOV    #BUFONE,RMBAO ;DATA BUFFER ADDRESS
1647 025270 012737 000063 001412  MOV    #WH!GO,RMCS10 ;WRITE HEADER AND DATA
1648
1649      ;VERIFY THAT SECTOR IS NOT BAD
      025276 004737 033152      JSR    PC,BADSCT     ;CALL BAD SECTOR MODULE
      025302 000405      BR     1$             ;GO TO 1$ IF NO ERROR
      025304 104401 063346      TYPE   ,SCTMSG      ;TYPE BAD SECTOR MESSAGE
      025310 104000      EMT                    ;ERROR # DEFINED BY BADSCT SUBROUTINE
      025312 000137 026116      JMP    17$           ;GO TO 17$ IF ERROR
1650 025316      1$:
1651 025316 012737 001420 001174  MOV    #RMDAO,$TMP0  ;USE SECTOR FOR DATA PATTERN
1652 025324 012737 000001 001176  MOV    #1,$TMP1
  
```



```

025522 104000          EMT          ;ERROR # DEFINED BY GET SUBROUTINE
025524 000137 026116  JMP          ;GO TO 17$ IF ERROR
1688 025530          6$:
1689
1690
1691 025530 004737 036766 ;VERIFY RESULTS OF WRITE COMMAND
      025534 000405  JSR          PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
      025536 000240  BR           7$          ;GO TO 7$ IF NO ERROR
      025540 104000  NOP          ;RETURN HERE IF ERROR
      025542 004736  EMT          ;ERROR # DEFINED BY PRIERR SUBROUTINE
      025544 000137 026116  JSR          PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      025550          JMP          17$          ;GO TO 17$ IF ERROR
1692 025550          7$:
1693 025550 004737 051502  JSR          PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      025554 000405  BR           8$          ;GO TO 8$ IF NO ERROR
      025556 000240  NOP          ;RETURN HERE IF ERROR
      025560 104000  EMT          ;ERROR # DEFINED BY DTASTS SUBROUTINE
      025562 004736  JSR          PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      025564 000137 026116  JMP          17$          ;GO TO 17$ IF ERROR
1694 025570          8$:
1695 025570 004737 037620  JSR          PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
      025574 000405  BR           9$          ;GO TO 9$ IF NO ERROR
      025576 000240  NOP          ;RETURN HERE IF ERROR
      025600 104000  EMT          ;ERROR # DEFINED BY SECERR SUBROUTINE
      025602 004736  JSR          PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      025604 000137 026116  JMP          17$          ;GO TO 17$ IF ERROR
1696 025610          9$:
1697
1698 025610 012737 000015 001412 ;OFFSET DEVICE FOR READ
      025616 012702 001553  MOV          #OFFSET!GO,RMCS10 ;CHANGE TO OFFSET COMMAND
1699 025616 012702 001553  MOV          #PUTINX,R2      ;WRITE PUT INDEX TABLE
1700 025622 112722 000006  MOVB        #RMDA,(R2)+
1701 025626 112722 000034  MOVB        #RMDC,(R2)+
1702 025632 112722 000032  MOVB        #RMOF,(R2)+
1703 025636 112722 000000  MOVB        #RMCS1,(R2)+
1704 025642 112722 000200  MOVB        #200,(R2)+
1705
1706 025646 004737 036240  JSR          PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      025652 000404  BR           10$         ;GO TO 10$ IF NO ERROR
      025654 000240  NOP          ;RETURN HERE IF ERROR
      025656 104000  EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
      025660 000137 026116  JMP          17$          ;GO TO 17$ IF ERROR
1707 025664          10$:
1708
1709
1710
1711 025664 012737 000073 001412 ;READ HEADER AND DATA AT OFFSET
      025672 012737 102364 001416  MOV          #RH!GO,RMCS10 ;READ HEADER & DATA COMMAND
      025700 012702 001553  MOV          #BUFTWO,RMBA0 ;CHANGE BUS ADDRESS
      025704 112722 000002  MOV          #PUTINX,R2      ;WRITE PUT INDEX TABLE
      025710 112722 000004  MOVB        #RMWC,(R2)+
      025714 112722 000000  MOVB        #RMBB,(R2)+
      025720 112722 000200  MOVB        #RMCS1,(R2)+
      025724 112722 000200  MOVB        #200,(R2)+
1718
1719 025724 004737 036240  JSR          PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      025730 000404  BR           11$         ;GO TO 11$ IF NO ERROR
      025732 000240  NOP          ;RETURN HERE IF ERROR
      025734 104000  EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
      025736 000137 026116  JMP          17$          ;GO TO 17$ IF ERROR

```



```

1720 025742          11$:
1721
1722
1723 025742 004737 036602      :WAIT FOR READ TO COMPLETE AND GET STATUS
1724          JSR      PC,TIMOUT      :WAIT FOR READ TO COMPLETE
1725 025746 004737 035770          JSR      PC,GET      :GO READ REGISTER(S) WITH GET SUBROUTINE
1726          025752 000404          BR      12$          :GO TO 12$ IF NO ERROR
1727          025754 000240          NOP          :RETURN HERE IF ERROR
1728          025756 104000          EMT          :ERROR # DEFINED BY GET SUBROUTINE
1729          025760 000137 026116  JMP      17$          :GO TO 17$ IF ERROR
1730
1731 025764 004737 036766      :VERIFY THE RESULTS OF READ OPERATION
1732          025770 000405          JSR      PC,PRIERR   :GO CHECK FOR PRIMARY ERRORS
1733          025772 000240          BR      13$          :GO TO 13$ IF NO ERROR
1734          025774 104000          NOP          :RETURN HERE IF ERROR
1735          025776 004736          EMT          :ERROR # DEFINED BY PRIERR SUBROUTINE
1736          026000 000137 026116  JSR      PC,@(SP)+   :GO BACK FOR MORE ERROR CHECKS
1737          026004          JMP      17$          :GO TO 17$ IF ERROR
1738
1739 026004 004737 051502      13$:
1740          026010 000405          JSR      PC,DTASTS   :GO VERIFY RESULTS OF DATA TRANSFER
1741          026012 000240          BR      14$          :GO TO 14$ IF NO ERROR
1742          026014 104000          NOP          :RETURN HERE IF ERROR
1743          026016 004736          EMT          :ERROR # DEFINED BY DTASTS SUBROUTINE
1744          026020 000137 026116  JSR      PC,@(SP)+   :GO BACK FOR MORE ERROR CHECKS
1745          026024          JMP      17$          :GO TO 17$ IF ERROR
1746
1747 026024 004737 037620      14$:
1748          026030 000405          JSR      PC,SECERR   :GO CHECK FOR SECONDARY ERRORS
1749          026032 000240          BR      15$          :GO TO 15$ IF NO ERROR
1750          026034 104000          NOP          :RETURN HERE IF ERROR
1751          026036 004736          EMT          :ERROR # DEFINED BY SECERR SUBROUTINE
1752          026040 000137 026116  JSR      PC,@(SP)+   :GO BACK FOR MORE ERROR CHECKS
1753          026044          JMP      17$          :GO TO 17$ IF ERROR
1754
1755
1756
1757 026044 004737 035342      :VERIFY DATA
1758          026050 101360          JSR      PC,CMPBUF   :GO COMPARE WRITE, READ DATA BUFFERS
1759          026052 102364          .WORD   BUFWO        :STARTING ADDRESS OF WRITE BUFFER
1760          026054 000404          .WORD   BUFTWO       :STARTING ADDRESS OF READ BUFFER
1761          026056 000240          BR      16$          :GO TO 16$ IF NO ERROR
1762          026060 104000          NOP          :RETURN HERE IF ERROR
1763          026062 000137 026116  EMT          :ERROR # DEFINED BY CMPBUF SUBROUTINE
1764          026066          JMP      17$          :GO TO 17$ IF ERROR
1765
1766 026066 032737 000200 001444 16$:
1767 026074 001010          BIT     #0,RMOFO     :DONE BOTH OFFSETS??
1768 026076 052737 000200 001444  BNE     #0,RMOFO     :YES!!
1769 026104 012737 101360 001416  BIS     #4,RMOFO     :SET FORWARD OFFSET
1770 026112 000137 025336          MOV     #4,JNE,RMBAO :CHANGE DATA BUFFER
1771 026116          JMP     2$
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
  
```

```

026116
026116 000004
026120 000240
  
```

```

*****
:*TEST 26          IVC FORMAT TEST
*****
TST26:
      SCOPE          :SCOPE CALL
      NOP           :START OF TEST
  
```

```

026122 012706 001100      MOV      #STACK,SP      ;INITIALIZE STACK POINTER
026126 013700 001276      MOV      $BASE,R0       ;R0 = UNIBUS ADDRESS
026132 013701 001466      MOV      TSTQUE,R1      ;(R1) = DEVICE BEING TESTED
026136 012737 000026 001226  MOV      #26,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

1747
1748
1749 026144 012737 000000 001446 ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
1750 026152 012737 000000 001420      MOV      #0,RMDCO       ;CYLINDER = 0
1751 026160 012737 010000 001444      MOV      #0,RMDAO       ;TRACK = 0, SECTOR = 0
1752 026166 012737 177376 001414      MOV      #FMT16,RMOFO   ;16 BIT FORMAT
1753 026174 012737 101360 001416      MOV      #-258.,RMWCO   ;2 + 256 WORDS (2'S COMP)
1754 026202 012737 000062 001412      MOV      #BUFONE,RMBAO  ;DATA BUFFER ADDRESS
1755
1756
026210 004737 033152      JSR      PC,BADSCT      ;CALL BAD SECTOR MODULE
026214 000405              BR       1$             ;GO TO 1$ IF NO ERROR
026216 104401 063346              TYPE     ,SCTMSG       ;TYPE BAD SECTOR MESSAGE
026222 104000              EMT
026224 000137 026740              JMP      16$           ;ERROR # DEFINED BY BADSCT SUBROUTINE
                                ;GO TO 16$ IF ERROR
1757 026230
1758 026230 012737 001420 001174 1$:      MOV      #RMDAO,$TMP0   ;USE SECTOR FOR DATA PATTERN
1759 026236 012737 000001 001176      MOV      #1,$TMP1
1760 026244 004737 035104      JSR      PC,GENBUF     ;GO GENERATE DATA BUFFER
1761 026250
1762
1763
1764 026250 004737 032176      ;PREPARE DEVICE FOR DATA TRANSFER
026254 154130      JSR      PC,TSTPRP    ;PREPARE DEVICE FOR TEST
                                ;TASK DESCRIPTOR AS FOLLOWS:
                                ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;CLEAR CONTROLLER & SELECT DEVICE
                                ;VERIFY CONTROLLER CLEAR OPERATION
                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;VERIFY PACK ACKNOWLEDGE
                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                ;VERIFY RECALIBRATION
                                ;GO TO 3$ IF NO ERROR
                                ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                ;GO TO 16$ IF ERROR
                                3$:
026256 000404              BR       3$
026260 000240              NOP
026262 104000              EMT
026264 000137 026740              JMP      16$
1765 026270
1766
1767
1768 026270 012737 000001 001436 ;RESET VOLUME VALID BY SETTING AND RESETTING DIAGNOSTIC MODE
1769 026276 112737 000024 001553      MOV      #DMD,RMMR10   ;SET DIAGNOSTIC MODE
1770 026304 112737 000200 001554      MOV      #RMMR1,PUTINX ;WRITE REGISTER INDEX TABLE
1771
1772 026312 004737 036240      JSR      PC,PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
026316 000404              BR       4$           ;GO TO 4$ IF NO ERROR
026320 000240              NOP
026322 104000              EMT
026324 000137 026740              JMP      16$           ;RETURN HERE IF ERROR
                                ;ERROR # DEFINED BY PUT SUBROUTINE
                                ;GO TO 16$ IF ERROR
1773 026330
1774
1775
1776 026330 012737 000063 001412 ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1777 026336 012737 000000 001436      MOV      #WH!GO,RMCS10 ;WRITE HEADER AND DATA
1778 026344 012702 001553      MOV      #0,RMMR10     ;RESET DIAGNOSTIC MODE
                                MOV      #PUTINX,R2    ;WRITE REGISTER INDEX TABLE

```

```

1779 026350 112722 000024      MOVB  #RMMR1,(R2)+
1780 026354 112722 000006      MOVB  #RMDA,(R2)+
1781 026360 112722 000034      MOVB  #RMDC,(R2)+
1782 026364 112722 000032      MOVB  #RMOF,(R2)+
1783 026370 112722 000002      MOVB  #RMMC,(R2)+
1784 026374 112722 000004      MOVB  #RMBB,(R2)+
1785 026400 112722 000000      MOVB  #RMCS1,(R2)+
1786 026404 112722 000200      MOVB  #200,(R2)+
1787
1788 026410 004737 036240      JSR   PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      026414 000404      BR    5$             ;GO TO 5$ IF NO ERROR
      026416 000240      NOP                    ;RETURN HERE IF ERROR
      026420 104000      EMT                    ;ERROR # DEFINED BY PUT SUBROUTINE
      026422 000137 026740      JMP   16$           ;GO TO 16$ IF ERROR
1789 026426      5$:
1790
1791      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1792 026426 004737 035704      JSR   PC,GETSTS
1793 026432 004737 036602      JSR   PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
1794
1795 026436 004737 035770      JSR   PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
      026442 000404      BR    6$             ;GO TO 6$ IF NO ERROR
      026444 000240      NOP                    ;RETURN HERE IF ERROR
      026446 104000      EMT                    ;ERROR # DEFINED BY GET SUBROUTINE
      026450 000137 026740      JMP   16$           ;GO TO 16$ IF ERROR
1796 026454      6$:
1797
1798      ;VERIFY RESULTS OF WRITE COMMAND
1799 026454 004737 036766      JSR   PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      026460 000405      BR    7$             ;GO TO 7$ IF NO ERROR
      026462 000240      NOP                    ;RETURN HERE IF ERROR
      026464 104000      EMT                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
      026466 004736      JSR   PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      026470 000137 026740      JMP   16$           ;GO TO 16$ IF ERROR
1800 026474      7$:
1801 026474 032737 010000 001400      BIT   #IVC,RMER2I    ;IS IVC STATUS SET??
1802 026502 001012      BNE   8$             ;YES!!
1803 026504 013737 001400 001142      MOV   RMER2I,$BDDAT ;RECEIVED STATUS FOR TYPEOUT
1804 026512 013737 001400 001140      MOV   RMER2I,$GDDAT ;EXPECTED STATUS
1805 026520 052737 010000 001140      BIS   #IVC,$GDDAT
1806 026526 104342      EMT   342
1807 026530      8$:
1808 026530 004737 037620      JSR   PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      026534 000405      BR    9$             ;GO TO 9$ IF NO ERROR
      026536 000240      NOP                    ;RETURN HERE IF ERROR
      026540 104000      EMT                    ;ERROR # DEFINED BY SECERR SUBROUTINE
      026542 004736      JSR   PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      026544 000137 026740      JMP   16$           ;GO TO 16$ IF ERROR
1809 026550      9$:
1810
1811      ;CLEAR IVC ERROR
1812 026550 004737 032176      JSR   PC,TSTPRP     ;PREPARE DEVICE FOR TEST
      026554 040100      .WORD 040100        ;TASK DESCRIPTOR AS FOLLOWS:
      ;CLEAR CONTROLLER & SELECT DEVICE
      ;VERIFY CONTROLLER CLEAR OPERATION
      026556 000404      BR    10$           ;GO TO 10$ IF NO ERROR
      026560 000240      NOP                    ;RETURN HERE IF ERROR

```

```

026562 104000          EMT          :ERROR # DEFINED BY TSTPRP SUBROUTINE
026564 000137 026740  JMP          16$          :GO TO 16$ IF ERROR
1813 026570          10$:
1814
1815          :READ HEADER AND DATA FOR SECTOR JUST WRITTEN
1816 026570 012737 000073 001412  MOV          #RH!GO,RMCS10 :READ HEADER & DATA COMMAND
1817 026576 012737 102364 001416  MOV          #BUFTWO,RMBAO :CHANGE BUS ADDRESS
1818
1819 026604 004737 036240  JSR          PC,PUT          :GO WRITE REGISTER(S) WITH PUT SUBROUTINE
026610 000404          BR          11$          :GO TO 11$ IF NO ERROR
026612 000240          NOP          :RETURN HERE IF ERROR
026614 104000          EMT          :ERROR # DEFINED BY PUT SUBROUTINE
026616 000137 026740  JMP          16$          :GO TO 16$ IF ERROR
1820 026622          11$:
1821
1822          :SETUP INPUT REGISTER BUFFER FOR READING STATUS
1823 026622 004737 035704  JSR          PC,GETSTS
1824 026626 004737 036602  JSR          PC,TIMOUT      :WAIT FOR READ TO COMPLETE
1825
1826 026632 004737 035770  JSR          PC,GET          :GO READ REGISTER(S) WITH GET SUBROUTINE
026636 000404          BR          12$          :GO TO 12$, IF NO ERROR
026640 000240          NOP          :RETURN HERE IF ERROR
026642 104000          EMT          :ERROR # DEFINED BY GET SUBROUTINE
026644 000137 026740  JMP          16$          :GO TO 16$ IF ERROR
1827 026650          12$:
1828
1829          :VERIFY THE RESULTS OF READ OPERATION
1830 026650 004737 036766  JSR          PC,PRIERR      :GO CHECK FOR PRIMARY ERRORS
026654 000405          BR          13$          :GO TO 13$ IF NO ERROR
026656 000240          NOP          :RETURN HERE IF ERROR
026660 104000          EMT          :ERROR # DEFINED BY PRIERR SUBROUTINE
026662 004736          JSR          PC,@(SP)+    :GO BACK FOR MORE ERROR CHECKS
026664 000137 026740  JMP          16$          :GO TO 16$ IF ERROR
1831 026670          13$:
1832 026670 032737 010000 001400  BIT          #IVC,RMER2I    :IS IVC STATUS SET??
1833 026676 001012          BNE          14$          :YES!!
1834 026700 013737 001400 001142  MOV          RMER2I,$BDDAT  :RECEIVED STATUS FOR TYPEOUT
1835 026706 013737 001400 001140*  MOV          RMER2I,$GDDAT  :EXPECTED STATUS
1836 026714 052737 010000 001140  BIS          #IVC,$GDDAT
1837 026722 104342          EMT          342
1838 026724          14$:
1839 026724 004737 037620  JSR          PC,SECERR      :GO CHECK FOR SECONDARY ERRORS
026730 000405          BR          15$          :GO TO 15$ IF NO ERROR
026732 000240          NOP          :RETURN HERE IF ERROR
026734 104000          EMT          :ERROR # DEFINED BY SECERR SUBROUTINE
026736 004736          JSR          PC,@(SP)+    :GO BACK FOR MORE ERROR CHECKS
1840 026740          15$:
1841
1842 026740          16$:
1843
1844
:*****
:*TEST 27          FORMAT ERROR TEST
:*****
TST27:
026740
026740 000004          SCOPE          :SCOPE CALL
026742 000240          NOP          :START OF TEST
026744 012706 001100  MOV          #STACK,SP     :INITIALIZE STACK POINTER
  
```

```

026750 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
026754 013701 001466      MOV      TSTQUE,R1     ;(R1) = DEVICE BEING TESTED
026760 012737 000027 001226  MOV      #27,$TESTN    ;:SET TEST NUMBER IN APT MAIL BOX

1845
1846      ;:SETUP PARAMETERS FOR GENERATING DATA BUFFER
1847 026766 012737 000000 001446  MOV      #0,RMDCO      ;CYLINDER = 0
1848 026774 012737 000000 001420  MOV      #0,RMDAO      ;TRACK = 0, SECTOR = 0
1849 027002 012737 000000 001444  MOV      #0,RMOFO      ;18 BIT FORMAT
1850 027010 012737 177376 001414  MOV      #-258.,RMWCO   ;2 + 256 WORDS (2'S COMP)
1851 027016
1852 027016 012737 101360 001416 1$:      MOV      #BUFONE,RMBA0 ;:DATA BUFFER ADDRESS
1853 027024 012737 000062 001412  MOV      #WH,RMCS10    ;:WRITE HEADER AND DATA
1854
1855      ;:VERIFY THAT SECTOR IS NOT BAD
027032 004737 033152      JSR      PC,BADSCT     ;:CALL BAD SECTOR MODULE
027036 000405              BR        2$           ;:GO TO 2$ IF NO ERROR
027040 104401 063346      TYPE     ,SCTMSG      ;:TYPE BAD SECTOR MESSAGE
027044 104000              EMT                    ;:ERROR # DEFINED BY BADSCT SUBROUTINE
027046 000137 027522      JMP      16$          ;:GO TO 16$ IF ERROR
1856 027052
1857 027052 012737 065000 001174 2$:      MOV      #ZEROS,$TMP0  ;:USE ALL ZEROS DATA PATTERN
1858 027060 012737 000001 001176  MOV      #1,$TMP1
1859 027066 004737 035104      JSR      PC,GFNBUFF   ;:GO GENERATE DATA BUFFER
1860 027072
1861
1862      ;:PREPARE DEVICE FOR DATA TRANSFER
1863 027072 004737 032176      JSR      PC,TSTPRP    ;:PREPARE DEVICE FOR TEST
027076 154130      .WORD   154130       ;:TASK DESCRIPTOR AS FOLLOWS:
                                ;:SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                ;:CLEAR CONTROLLER & SELECT DEVICE
                                ;:VERIFY CONTROLLER CLEAR OPERATION
                                ;:PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;:VERIFY PACK ACKNOWLEDGE
                                ;:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                ;:VERIFY RECALIBRATION
027100 000404              BR        4$           ;:GO TO 4$ IF NO ERROR
027102 000240              NOP                    ;:RETURN HERE IF ERROR
027104 104000              EMT                    ;:ERROR # DEFINED BY TSTPRP SUBROUTINE
027106 000137 027522      JMP      16$          ;:GO TO 16$ IF ERROR
1864 027112
1865
1866      ;:SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1867 027112 012737 000063 001412  MOV      #WH!GO,RMCS10 ;:WRITE HEADER AND DATA
1868 027120 012702 001553      MOV      #PUTINX,R2    ;:WRITE REGISTER INDEX TABLE
1869 027124 112722 000006      MOV      #RMDA,(R2)+
1870 027130 112722 000034      MOV      #RMDC,(R2)+
1871 027134 112722 000032      MOV      #RMOF,(R2)+
1872 027140 112722 000002      MOV      #RMWC,(R2)+
1873 027144 112722 000004      MOV      #RMBA,(R2)+
1874 027150 112722 000000      MOV      #RMCS1,(R2)+
1875 027154 112722 000200      MOV      #200,(R2)+
1876
1877 027160 004737 036240      JSR      PC,PUT        ;:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
027164 000404              BR        5$           ;:GO TO 5$ IF NO ERROR
027166 000240              NOP                    ;:RETURN HERE IF ERROR
027170 104000              EMT                    ;:ERROR # DEFINED BY PUT SUBROUTINE
027172 000137 027522      JMP      16$          ;:GO TO 16$ IF ERROR
    
```

```

1878 027176      5$:
1879
1880      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1881 027176 004737 035704      JSR      PC,GETSTS
1882 027202 004737 036602      JSR      PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
1883
1884 027206 004737 035770      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      027212 000404      BR      6$      ;GO TO 6$ IF NO ERROR
      027214 000240      NOP      ;RETURN HERE IF ERROR
      027216 104000      EMT      ;ERROR # DEFINED BY GET S BRROUTINE
      027220 000137 027522      JMP      16$      ;GO TO 16$ IF ERROR
1885 027224
1886
1887      ;VERIFY RESULTS OF WRITE COMMAND
1888 027224 004737 036766      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      027230 000405      BR      7$      ;GO TO 7$ IF NO ERROR
      027232 000240      NOP      ;RETURN HERE IF ERROR
      027234 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      027236 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      027240 000137 027522      JMP      16$      ;GO TO 16$ IF ERROR
1889 027244
1890 027244 004737 051502      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      027250 000405      BR      8$      ;GO TO 8$ IF NO ERROR
      027252 000240      NOP      ;RETURN HERE IF ERROR
      027254 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      027256 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      027260 000137 027522      JMP      16$      ;GO TO 16$ IF ERROR
1891 027264
1892 027264 004737 037620      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      027270 000405      BR      9$      ;GO TO 9$ IF NO ERROR
      027272 000240      NOP      ;RETURN HERE IF ERROR
      027274 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      027276 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      027300 000137 027522      JMP      16$      ;GO TO 16$ IF ERROR
1893 027304
1894
1895      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN IN OPPOSITE FORMAT
1896      ;TO CHECK THAT "FER" SETS
1897 027304 005137 001444      COM      RMOFO      ;CHANGE TO OPPOSITE FORMAT
1898 027310 042737 167777 001444      BIC      #^C<FMT16>,RMOFO
1899 027316 012737 102364 001416      MOV      #BUFTWO,RMBAO      ;CHANGE BUS ADDRESS
1900 027324 012737 000073 001412      MOV      #RH!GO,RMCS10      ;READ HEADER & DATA COMMAND
1901
1902 027332 004737 036240      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      027336 000404      BR      10$      ;GO TO 10$ IF NO ERROR
      027340 000240      NOP      ;RETURN HERE IF ERROR
      027342 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      027344 000137 027522      JMP      16$      ;GO TO 16$ IF ERROR
1903 027350
1904
1905      ;WAIT FOR READ TO COMPLETE AND GET STATUS
1906 027350 004737 036602      JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
1907
1908 027354 004737 035770      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      027360 000404      BR      11$      ;GO TO 11$ IF NO ERROR
      027362 000240      NOP      ;RETURN HERE IF ERROR
      027364 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
  
```

```

1909 027366 000137 027522      JMP      16$      ;GO TO 16$ IF ERROR
1910 027372
1911
1912 027372 004737 036766      ;VERIFY THE RESULTS OF READ OPERATION
      JSR      PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
      BR       12$      ;GO TO 12$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP      16$      ;GO TO 16$ IF ERROR
1913 027412
1914
1915
1916 027412 032737 000020 001352 ;VERIFY THAT FORMAT ERROR IS SET
      BIT      #FER,RMER11 ;IS FORMAT ERROR SET??
      BNE      14$      ;YES..
1917 027420 001022
1918
1919 027422 004737 051502      JSR      PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
      BR       13$      ;GO TO 13$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP      16$      ;GO TO 16$ IF ERROR
1920
1921 027442 013737 001352 001142 13$:  MOV      RMER11,$BDDAT ;BAD DATA FOR TYPEOUT
1922 027450 013737 001352 001140      MOV      RMER11,$GDDAT ;EXPECTED DATA
1923 027456 052737 000020 001140      BIS      #FER,$GDDAT
1924 027464 104343      EMT      343
1925 027466
1926 027466 004737 037620      14$:  JSR      PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
      BR       15$      ;GO TO 15$ IF NO ERROR
      NOP      ;RETURN HERE IF ERROR
      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
      JMP      16$      ;GO TO 16$ IF ERROR
1927 027506
1928 027506 032737 010000 001444 15$:  BIT      #FMT16,RMOFO ;TEST WRITE 16 BIT FORMAT AND
1929                                ;READ 18 BIT MODE ?
1930 027514 001402      BEQ      16$      ;BR IF YES
1931 027516 000137 027016      JMP      1$       ;TEST AGAIN
1932 027522
1933
1934
;*****
;*TEST 30          FORMAT HCE, FIRST HEADER WORD
;*****
TST30:
      SCOPE      ;SCOPE CALL
      NOP      ;START OF TEST
      MOV      #STACK,SP ;INITIALIZE STACK POINTER
      MOV      $BASE,R0  ;R0 = UNIBUS ADDRESS
      MOV      TSTQUE,R1 ;(R1) = DEVICE BEING TESTED
      MC /      #30,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
1935
1936 027550 012737 000001 030520      MOV      #1,23$ ;INIT BIT POSITION
1937
1938
1939 027556      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
1940 027556 012737 010000 001444 1$:  MOV      #FMT16,RMOFO ;16 BIT FORMAT

```

1941	027564	012737	000000	001446	MOV	#0,RMDCO	:CYLINDER - 0
1942	027572	012737	000000	001420	MOV	#0,RMDAO	:TRACK = 0, SECTOR = 0
1943	027600	012737	177376	001414	MOV	#-258.,RMWCO	:2 + 256 WORDS (2'S COMP)
1944	027606	012737	101360	001416	MOV	#BUFONE,RMBAO	:DATA BUFFER ADDRESS
1945	027614	012737	000062	001412	MOV	#WH,RMCS10	:WRITE HEADER AND DATA
1946							
1947							:VERIFY THAT SECTOR IS NOT BAD
	027622	004737	033152		JSR	PC,BADSCT	:CALL BAD SECTOR MODULE
	027626	000405			BR	2\$:GO TO 2\$ IF NO ERROR
	027630	104401	063346		TYPE	,SCTMSG	:TYPE BAD SECTOR MESSAGE
	027634	104000			EMT		:ERROR # DEFINED BY BADSCT SUBROUTINE
	027636	000137	030672		JMP	27\$:GO TO 27\$ IF ERROR
1948	027642						2\$:
1949	027642	012737	065000	001174	MOV	#ZEROS,\$TMP0	:USE ALL ZEROS DATA PATTERN
1950	027650	012737	000001	001176	MOV	#1,\$TMP1	
1951	027656	004737	035104		JSR	PC,GENBUF	:GO GENERATE DATA BUFFER
1952	027662						3\$:
1953							
1954							:PREPARE DEVICE FOR DATA TRANSFER
1955	027662	004737	032176		JSR	PC,TSTPRP	:PREPARE DEVICE FOR TEST
	027666	154130			.WORD	154130	:TASK DESCRIPTOR AS FOLLOWS:
							:SELECT DEVICE & VERIFY DEVICE AVAILABLE
							:CLEAR CONTROLLER & SELECT DEVICE
							:VERIFY CONTROLLER CLEAR OPERATION
							:PACK ACKNOWLEDGE IF VOLUME NOT VALID
							:VERIFY PACK ACKNOWLEDGE
							:RECALIBRATE IF 'SKI' OR 'PIP' IS SET
							:VERIFY RECALIBRATION
	027670	000404			BR	4\$:GO TO 4\$ IF NO ERROR
	027672	000240			NOP		:RETURN HERE IF ERROR
	027674	104000			EMT		:ERROR # DEFINED BY TSTPRP SUBROUTINE
	027676	000137	030672		JMP	27\$:GO TO 27\$ IF ERROR
1956	027702						4\$:
1957							
1958							:COMPLEMENT DATA BIT IN FIRST HEADER WORD
1959	027702	033737	030520	101360	BIT	23\$,BUFONE	:IS BIT IN HEADER ON??
1960	027710	001404			BEQ	5\$:NO!!
1961	027712	043737	030520	101360	BIC	23\$,BUFONE	:RESET BIT IN HEADER
1962	027720	000403			BR	6\$	
1963	027722	053737	030520	101360	5\$:	BIS	23\$,BUFONE
1964							:SET BIT IN HEADER
1965							:SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
1966	027730						6\$:
1967	027730	012737	000063	001412	MOV	#WH!GO,RMCS10	:WRITE HEADER AND DATA
1968	027736	012702	001553		MOV	#PUTINX,R2	:WRITE REGISTER INDEX ABLE
1969	027742	112722	000006		MOVB	#RMDA,(R2)+	
1970	027746	112722	000034		MOVB	#RMDC,(R2)+	
1971	027752	112722	000032		MOVB	#RMOF,(R2)+	
1972	027756	112722	000002		MOVB	#RMWC,(R2)+	
1973	027762	112722	000004		MOVB	#RMB A,(R2)+	
1974	027766	112722	000000		MOVB	#RMCS1,(R2)+	
1975	027772	112722	000200		MOVB	#200,(R2)+	
1976							
1977	027776	004737	036240		JSR	PC,PUT	:GO WRITE REGISTER(S) WITH PUT SUBROUTINE
	030002	000404			BR	7\$:GO TO 7\$ IF NO ERROR
	030004	000240			NOP		:RETURN HERE IF ERROR
	030006	104000			EMT		:ERROR # DEFINED BY PUT SUBROUTINE


```

1978 030010 000137 030516      7$:      JMP      22$      ;GO TO 22$ IF ERROR
1979 030014
1980 ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
1981 030014 004737 035704      JSR      PC,GETSTS
1982 030020 004737 036602      JSR      PC,TIMOUT      ;WAIT FOR COMMAND TO COMPLETE
1983
1984 030024 004737 035770      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      030030 000404      BR      8$      ;GO TO 8$ IF NO ERROR
      030032 000240      NOP      ;RETURN HERE IF ERROR
      030034 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      030036 000137 030516      JMP      22$      ;GO TO 22$ IF ERROR
1985 030042      8$:
1986
1987 ;VERIFY RESULTS OF WRITE COMMAND
1988 030042 004737 036766      JSR      PC,PRIERR      ;GO CHECK FOR PRIMARY ERRORS
      030046 000405      BR      9$      ;GO TO 9$ IF NO ERROR
      030050 000240      NOP      ;RETURN HERE IF ERROR
      030052 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
      030054 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      030056 000137 030516      JMP      22$      ;GO TO 22$ IF ERROR
1989 030062      9$:
1990 030062 004737 051502      JSR      PC,DTASTS      ;GO VERIFY RESULTS OF DATA TRANSFER
      030066 000405      BR      10$      ;GO TO 10$ IF NO ERROR
      030070 000240      NOP      ;RETURN HERE IF ERROR
      030072 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
      030074 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      030076 000137 030516      JMP      22$      ;GO TO 22$ IF ERROR
1991 030102      10$:
1992 030102 004737 037620      JSR      PC,SECERR      ;GO CHECK FOR SECONDARY ERRORS
      030106 000405      BR      11$      ;GO TO 11$ IF NO ERROR
      030110 000240      NOP      ;RETURN HERE IF ERROR
      030112 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
      030114 004736      JSR      PC,@(SP)+      ;GO BACK FOR MORE ERROR CHECKS
      030116 000137 030516      JMP      22$      ;GO TO 22$ IF ERROR
1993 030122      11$:
1994
1995 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
1996 030122 012737 000073 001412      MOV      #RH.GO,RMCS10      ;READ HEADER & DATA COMMAND
1997 030130 012737 102364 001416      MOV      #BUFTWO,RMBAO      ;CHANGE BUS ADDRESS
1998
1999 030136 004737 036240      JSR      PC,PUT      ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
      030142 000404      BR      12$      ;GO TO 12$ IF NO ERROR
      030144 000240      NOP      ;RETURN HERE IF ERROR
      030146 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
      030150 000137 030516      JMP      22$      ;GO TO 22$ IF ERROR
2000 030154      12$:
2001
2002 ;WAIT FOR READ TO COMPLETE AND GET STATUS
2003 030154 004737 036602      JSR      PC,TIMOUT      ;WAIT FOR READ TO COMPLETE
2004
2005 030160 004737 035770      JSR      PC,GET      ;GO READ REGISTER(S) WITH GET SUBROUTINE
      030164 000404      BR      13$      ;GO TO 13$ IF NO ERROR
      030166 000240      NOP      ;RETURN HERE IF ERROR
      030170 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
      030172 000137 030516      JMP      22$      ;GO TO 22$ IF ERROR
2006 030176      13$:

```

FORMATTED FILE, FIRST HEADER WORD

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

030176 004737 036766
030202 000405
030204 000240
030206 104000
030210 004736
030212 000137 030516
030216
030216 032737 140000 030520
030224 001033
030226 032737 010000 030520
030234 001456
030236 032737 000C20 001352
030244 001107
030246 004737 051502
030252 000405
030254 000240
030256 104000
030260 004736
030262 000137 030516
030266
030266 013737 001352 001142
030274 013737 001352 001140
030302 052737 000020 001140
030310 104343
030312 000501
030314
030314 032737 100000 001400
030322 001060
030324 004737 051502
030330 000405
030332 000240
030334 104000
030336 004736
030340 000137 030516
030344
030344 013737 001400 001142
030352 013737 001400 001140
030360 052737 100000 001140
030366 104345
030370 000452
030372
030372 032737 000200 001352
030400 001031
030402 004737 051502
030406 000405
030410 000240

:VERIFY THE RESULTS OF READ OPERATION

JSR PC,PRIERR
BR 14\$
NOP
EMT
JSR PC,@(SP)+
JMP 22\$

:GO CHECK FOR PRIMARY ERRORS
:GO TO 14\$ IF NO ERROR
:RETURN HERE IF ERROR
:ERROR # DEFINED BY PRIERR SUBROUTINE
:GO BACK FOR MORE ERROR CHECKS
:GO TO 22\$ IF ERROR

14\$:

BIT #MSF!USE,23\$
BNE 16\$
BIT #FMT16,23\$
BEQ 18\$

:SHOULD BSE BE SET ?
:YES !!
:IS THIS FER ??
:NO !!

:VERIFY THAT FER IS SET

BIT #FER,RMER11
BNE 20\$

:IS FER SET ??
:YES !!

15\$:

JSR PC,DTASTS
BR 15\$
NOP
EMT
JSR PC,@(SP)+
JMP 22\$

:GO VERIFY RESULTS OF DATA TRANSFER
:GO TO 15\$ IF NO ERROR
:RETURN HERE IF ERROR
:ERROR # DEFINED BY DTASTS SUBROUTINE
:GO BACK FOR MORE ERROR CHECKS
:GO TO 22\$ IF ERROR

MOV RMER11,\$BDDAT
MOV RMER11,\$GDDAT
BIS #FER,\$GDDAT
EMT 343
BR 22\$

:RECEIVED STATUS
:EXPECTED STATUS

:VERIFY THAT BSE IS SET

16\$:

BIT #BSE,RMER21
BNE 20\$

:IS BSE SET ??
:YES !!

17\$:

JSR PC,DTASTS
BR 17\$
NOP
EMT
JSR PC,@(SP)+
JMP 22\$

:GO VERIFY RESULTS OF DATA TRANSFER
:GO TO 17\$ IF NO ERROR
:RETURN HERE IF ERROR
:ERROR # DEFINED BY DTASTS SUBROUTINE
:GO BACK FOR MORE ERROR CHECKS
:GO TO 22\$ IF ERROR

MOV RMER21,\$BDDAT
MOV RMER21,\$GDDAT
BIS #BSE,\$GDDAT
EMT 345
BR 22\$

:RECEIVED STATUS
:EXPECTED STATUS
:SKIP REST OF TEST

:VERIFY THAT HCE IS SET

18\$:

BIT #HCE,RMER11
BNE 20\$

:IS 'HCE' SET??
:YES.!

JSR PC,DTASTS
BR 19\$
NOP

:GO VERIFY RESULTS OF DATA TRANSFER
:GO TO 19\$ IF NO ERROR
:RETURN HERE IF ERROR

```

030412 104000          EMT
030414 004736          JSR
030416 000137 030516  JMP          PC,@(SP)+
                                22$          :ERROR # DEFINED BY DTASTS SUBROUTINE
                                :GO BACK FOR MORE ERROR CHECKS
                                :GO TO 22$ IF ERROR
2047 030422          19$:
2048 030422 013737 001352 001142  MOV          RMER11,$BDDAT          :RECEIVED STATUS FOR TYPEOUT
2049 030430 013737 001352 001140  MOV          RMER11,$GDDAT          :EXPECTED STATUS
2050 030436 052737 000200 001140  BIS          #HCE,$GDDAT
2051 030444 012737 000001 001174  MOV          #1,$TMP0
2052 030452 013737 030520 001176  MOV          23$,$TMP1          :GET HEADER WORD NUMBER
2053 030460 104344          EMT          34$          :GET FAILING BIT POSITION
2054 030462 000415          BR          22$
2055 030464          20$:
2056
2057          :CHECK FOR OTHER ERRORS
2058 030464 004737 037620  JSR          PC,SECERR          :GO CHECK FOR SECONDARY ERRORS
                                BR          21$          :GO TO 21$ IF NO ERROR
                                030472 000240  NOP          :RETURN HERE IF ERROR
                                030474 104000  EMT          :ERROR # DEFINED BY SECERR SUBROUTINE
                                030476 004736  JSR          PC,@(SP)+          :GO BACK FOR MORE ERROR CHECKS
                                C30500 000137 030516  JMP          22$          :GO TO 22$ IF ERROR
2059 030504          21$:
2060
2061          :ADVANCE THE BIT POSITION AND FORMAT AGAIN IF NOT DONE
2062 030504 006337 030520  ASL          23$          :SHIFT TO NEXT BIT POSITION
2063 030510 001404          BEQ          24$          :EXIT IF DONE
2064 030512 000137 027556  JMP          1$          :GO DO NEXT SECTOR
2065
2066 030516 000465          22$: BR          27$          :JUMP TO NEXT TEST
2067 030520 000000          23$: .WORD          0          :STORAGE FOR BIT POSITION
2068
2069          :*****
2070          :*REFORMAT SECTOR THAT WAS WRITTEN WITH BAD HEADER
2071          :*****
2072 030522          24$:
2073 030522 012737 010000 001444  MOV          #FMT16,RMOFO          :16 BIT MODE
2074 030530 012737 177776 001414  MOV          #-2,RMWCO          :ONLY TWO HEADER WORDS
2075 030536 012737 101360 001416  MOV          #BUFONE,RMBAO          :SELECT THE BUFFER
2076 030544 012737 000062 001412  MOV          #WH,RMCS10          :WRITE HEAD AND DATA COMMAND
2077 030552 004737 035104          JSR          PC,GENBUF          :GENERATE THE BUFFER PATTERN
2078
2079 030556 004737 032176          JSR          PC,TSTPRP          :PREPARE DEVICE FOR TEST
                                C30562 154130  .WORD          154130          :TASK DESCRIPTOR AS FOLLOWS:
                                :SELECT DEVICE & VERIFY DEVICE AVAILABLE
                                :CLEAR CONTROLLER & SELECT DEVICE
                                :VERIFY CONTROLLER CLEAR OPERATION
                                :PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                :VERIFY PACK ACKNOWLEDGE
                                :RECALIBRATE IF 'SKI' OR 'PIP' IS SET
                                :VERIFY RECALIBRATION
                                030564 000404          BR          25$          :GO TO 25$ IF NO ERROR
                                030566 000240          NOP          :RETURN HERE IF ERROR
                                030570 104000          EMT          :ERROR # DEFINED BY TSTPRP SUBROUTINE
                                030572 000137 030576  JMP          25$          :GO TO 25$ IF ERROR
2080 030576          25$:
2081 030576 012737 000063 001412  MOV          #WH!GO,RMCS10          :SET THE GO BIT
2082 030604 012702 001553          MOV          #PUTINX,R2          :PUT THE REGISTER ADDRESS INTO TABLE
2083 030610 112722 000006          MOVB         #RMDA,(R2)+
    
```

```

2084 030614 112722 000034      MOVB  #RMDC,(R2)+
2085 030620 112722 000032      MOVB  #RMOF,(R2)+
2086 030624 112722 000002      MOVB  #RMWC,(R2)+
2087 030630 112722 000004      MOVB  #RMBA,(R2)+
2088 030634 112722 000000      MOVB  #RMCS1,(R2)+
2089 030640 112722 000200      MCVB  #200,(R2)+
2090 030644 004737 036240      JSR   PC,PUT
      030650 000404      BR    26$
      030652 000240      NOP
      030654 104000      EMT
      030656 000137 030672      JMP   27$
2091 030662 000240      ;
2092 030662 000240      ;
2093 030664 004737 036602      JSR   PC,TIMOUT
2094 030670 000240      NOP
2095 030672
2096
2097

```

```

: TERMINATOR
: GO WRITE REGISTER(S) WITH PUT SUBROUTINE
: GO TO 26$ IF NO ERROR
: RETURN HERE IF ERROR
: ERROR # DEFINED BY PUT SUBROUTINE
: GO TO 27$ IF ERROR

```

```

26$:
NOP
JSR PC,TIMOUT ;WAIT UNTIL IT FINISH
NOP

```

27\$:

```

:*****
: *TEST 31 FORMAT HCE, SECOND HEADER WORD
:*****
TST31:

```

```

      030672
      030672 000004      SCOPE
      030674 000240      NOP
      030676 012706 001100      MOV   #STACK,SP
      030702 013700 001276      MOV   $BASE,R0
      030706 013701 001466      MOV   TSTQUE,R1
      030712 012737 000031 001226      MOV   #31,$TESTN
2098
2099 030720 012737 000001 031514      MOV   #1,19$
2100
2101      ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
2102 030726
2103 030726 012737 000000 001446      MOV   #0,RMDCO
2104 030734 012737 000000 001420      MOV   #0,RMDAO
2105 030742 012737 010000 001444      MOV   #FMT16,RMOFO
2106 030750 012737 177376 001414      MOV   #-258,RMWCO
2107 030756 012737 101360 001416      MOV   #BUFONE,RMBAO
2108 030764 012737 000062 001412      MOV   #WH,RMCS10
2109
2110      ;VERIFY THAT SECTOR IS NOT BAD
      030772 004737 033152      JSR   PC,BADSCT
      030776 000405      BR    2$
      031000 104401 063346      TYPE  ,SCTMSG
      031004 104000      EMT
      031006 000137 031666      JMP   23$
2111 031012
2112 031012 012737 065000 001174      MOV   #ZEROS,$TMP0
2113 031020 012737 000001 001176      MOV   #1,$TMP1
2114 031026 004737 035104      JSR   PC,GENBUF
2115 031032
2116
2117
2118 031032 004737 032176      ;PREPARE DEVICE FOR DATA TRANSFER
      031036 154130      JSR   PC,TSTPRP
      .WORD 154130

```

```

:SCOPE CALL
:START OF TEST
:INITIALIZE STACK POINTER
:R0 = UNIBUS ADDRESS
:(R1) = DEVICE BEING TESTED
::SET TEST NUMBER IN APT MAIL BOX
:INIT BIT POSITION
:CYLINDER = 0
:TRACK = 0, SECTOR = 0
:16 BIT FORMAT
:2 + 256 WORDS (2'S COMP)
:DATA BUFFER ADDRESS
:WRITE HEADER AND DATA
:CALL BAD SECTOR MODULE
:GO TO 2$ IF NO ERROR
:TYPE BAD SECTOR MESSAGE
:ERROR # DEFINED BY BADSCT SUBROUTINE
:GO TO 23$ IF ERROR
:USE ALL ZEROS DATA PATTERN
:GO GENERATE DATA BUFFER
:PREPARE DEVICE FOR TEST
:TASK DESCRIPTOR AS FOLLOWS:
:SELECT DEVICE & VERIFY DEVICE AVAILABLE
:CLEAR CONTROLLER & SELECT DEVICE
:VERIFY CONTROLLER CLEAR OPERATION

```

```

031040 000404          BR      4$          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
031042 000240          NOP          ;VERIFY PACK ACKNOWLEDGE
031044 104000          EMT          ;RECALIBRATE IF 'SKI' OR 'PIP' IS SET
031046 000137 031666  JMP      23$         ;VERIFY RECALIBRATION
;GO TO 4$ IF NO ERROR
;RETURN HERE IF ERROR
;ERROR # DEFINED BY TSTPRP SUBROUTINE
;GO TO 23$ IF ERROR

2119
2120          ;COMPLEMENT DATA BIT IN SECOND HEADER WORD
2121 031052          4$:
2122 031052 033737 031514 101362  BIT      19$,BUFONE+2  ;IS BIT IN HEADER ON??
2123 031060 007404          BEQ      5$          ;NO!
2124 031062 043737 031514 101362  BIC      19$,BUFONE+2  ;RESET BIT IN HEADER
2125 031070 000403          BR      6$          ;
2126 031072 053737 031514 101362  5$:  BIS      19$,BUFONE+2  ;SET BIT IN HEADER
2127
2128          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
2129 031100          6$:
2130 031100 012737 0C0063 001412  MOV      #WH.GO, RMCS10 ;WRITE HEADER AND DATA
2131 031106 012702 001553          MOV      #PUTINX, R2    ;WRITE REGISTER INDEX TABLE
2132 031112 112722 000006          MOVB    #RMDA, (R2)+
2133 031116 112722 000034          MOVB    #RMDC, (R2)+
2134 031122 112722 000032          MOVB    #RMOF, (R2)+
2135 031126 112722 000002          MOVB    #RMWC, (R2)+
2136 031132 112722 000004          MOVB    #RMBA, (R2)+
2137 031136 112722 000000          MOVB    #RMCS1, (R2)+
2138 031142 112722 000200          MOVB    #200, (R2)+
2139
2140 031146 004737 036240          JSR     PC, PUT        ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
031152 000404          BR      7$          ;GO TO 7$ IF NO ERROR
031154 000240          NOP          ;RETURN HERE IF ERROR
031156 104000          EMT          ;ERROR # DEFINED BY PUT SUBROUTINE
031160 000137 031512  JMP      18$         ;GO TO 18$ IF ERROR
2141 031164          7$:
2142
2143          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
2144 031164 004737 035704          JSR     PC, GETSTS
2145 031170 004737 036602          JSR     PC, TIMEOUT    ;WAIT FOR COMMAND TO COMPLETE
2146
2147 031174 004737 035770          JSR     PC, GET        ;GO READ REGISTER(S) WITH GET SUBROUTINE
031200 000404          BR      8$          ;GO TO 8$ IF NO ERROR
031202 000240          NOP          ;RETURN HERE IF ERROR
031204 104000          EMT          ;ERROR # DEFINED BY GET SUBROUTINE
031206 000137 031512  JMP      18$         ;GO TO 18$ IF ERROR
2148 031212          8$:
2149
2150          ;VERIFY RESULTS OF WRITE COMMAND
2151 031212 004737 036766          JSR     PC, PRIERR.    ;GO CHECK FOR PRIMARY ERRORS
031216 000405          BR      9$          ;GO TO 9$ IF NO ERROR
031220 000240          NOP          ;RETURN HERE IF ERROR
031222 104000          EMT          ;ERROR # DEFINED BY PRIERR SUBROUTINE
031224 004736          JSR     PC, @(SP)+    ;GO BACK FOR MORE ERROR CHECKS
031226 000137 031512  JMP      18$         ;GO TO 18$ IF ERROR
2152 031232          9$:
2153 031232 004737 051502          JSR     PC, DTASTS    ;GO VERIFY RESULTS OF DATA TRANSFER
031236 000405          BR      10$         ;GO TO 10$ IF NO ERROR

```

```

031240 000240      NOP      ;RETURN HERE IF ERROR
031242 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
031244 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031246 000137 031512 JMP      18$      ;GO TO 18$ IF ERROR
2154 031252      10$:
2155 031252 004737 037620 JSR      PC,SECERR ;GO CHECK FOR SECONDARY ERRORS
031256 000405      BR      11$      ;GO TO 11$ IF NO ERROR
031260 000240      NOP      ;RETURN HERE IF ERROR
031262 104000      EMT      ;ERROR # DEFINED BY SECERR SUBROUTINE
031264 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031266 000137 031512 JMP      18$      ;GO TO 18$ IF ERROR
2156 031272      11$:
2157
2158 ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
2159 031272 012737 000073 001412 MOV      #RH.GO,RMCS10 ;READ HEADER & DATA COMMAND
2160 031300 012737 102364 001416 MOV      #BUFTWO,RMBAO ;CHANGE BUS ADDRESS
2161
2162 031306 004737 036240 JSR      PC,PUT    ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
031312 000404      BR      12$      ;GO TO 12$ IF NO ERROR
031314 000240      NOP      ;RETURN HERE IF ERROR
031316 104000      EMT      ;ERROR # DEFINED BY PUT SUBROUTINE
031320 000137 031512 JMP      18$      ;GO TO 18$ IF ERROR
2163 031324      12$:
2164
2165 ;WAIT FOR READ TO COMPLETE AND GET STATUS
2166 031324 004737 036602 JSR      PC,TIMOUT ;WAIT FOR READ TO COMPLETE
2167
2168 031330 004737 035770 JSR      PC,GET    ;GO READ REGISTER(S) WITH GET SUBROUTINE
031334 000404      BR      13$      ;GO TO 13$ IF NO ERROR
031336 000240      NOP      ;RETURN HERE IF ERROR
031340 104000      EMT      ;ERROR # DEFINED BY GET SUBROUTINE
031342 000137 031512 JMP      18$      ;GO TO 18$ IF ERROR
2169 031346      13$:
2170
2171 ;VERIFY THE RESULTS OF READ OPERATION
2172 031346 004737 036766 JSR      PC,PRIERR ;GO CHECK FOR PRIMARY ERRORS
031352 000405      BR      14$      ;GO TO 14$ IF NO ERROR
031354 000240      NOP      ;RETURN HERE IF ERROR
031356 104000      EMT      ;ERROR # DEFINED BY PRIERR SUBROUTINE
031360 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031362 000137 031512 JMP      18$      ;GO TO 18$ IF ERROR
2173 031366      14$:
2174
2175 ;VERIFY THAT HEADER COMPARE ERROR IS SET
2176 031366 032737 000200 001352 BIT      #HCE,RMER1I ;IS 'HCE' SET??
2177 031374 001031      BNE     16$      ;YES!!
2178
2179 031376 004737 051502 JSR      PC,DTASTS ;GO VERIFY RESULTS OF DATA TRANSFER
031402 000405      BR      15$      ;GO TO 15$ IF NO ERROR
031404 000240      NOP      ;RETURN HERE IF ERROR
031406 104000      EMT      ;ERROR # DEFINED BY DTASTS SUBROUTINE
031410 004736      JSR      PC,@(SP)+ ;GO BACK FOR MORE ERROR CHECKS
031412 000137 031512 JMP      18$      ;GO TO 18$ IF ERROR
2180 031416      15$:
2181 031416 013737 001352 001142 MOV      RMER1I,$BDDAT ;RECEIVED STATUS FOR TYPEOUT
2182 031424 013737 001352 001140 MOV      RMER1I,$GDDAT ;EXPECTED STATUS
2183 031432 052737 000200 001140 BIS      #HCE,$GDDAT

```

```

2184 031440 012737 000002 001174      MOV      #2,$TMP0      ;GET HEADER WORD NUMBER
2185 031446 013737 031514 001176      MOV      19$,$TMP1     ;GET FAILING BIT POSITION
2186 031454 104344                      EMT      344
2187 031456 000415                      BR       18$
2188 031460
2189
2190
2191 031460 004737 037620      ;CHECK FOR OTHER ERRORS
      031464 000405      JSR      PC,SECERR     ;GO CHECK FOR SECONDARY ERRORS
      031466 000240      BR       17$          ;GO TO 17$ IF NO ERROR
      031470 104000      NOP                      ;RETURN HERE IF ERROR
      031472 004736      EMT                      ;ERROR # DEFINED BY SECERR SUBROUTINE
      031474 000137 031512 JSR      PC,@(SP)+     ;GO BACK FOR MORE ERROR CHECKS
2192 031500                      JMP      18$          ;GO TO 18$ IF ERROR
2193
2194
2195 031500 006337 031514      ;ADVANCE THE BIT POSITION AND FORMAT NEXT SECTOR IF NOT DONE
      031504 001404      ASL      19$          ;SHIFT TO NEXT BIT POSITION
2196 031506 000137 030726      BEQ      20$          ;EXIT IF DONE
2197 031506 000137 030726      JMP      1$           ;GO DO NEXT SECTOR
2198
2199 031512 000465      18$: BR       23$      ;JUMP OVER STORAGE
2200 031514 000000      19$: .WORD 0          ;STORAGE FOR BIT POSITION
2201
2202
2203
2204
2205 031516
2206 031516 012737 010000 001444      ;*****
2207 031524 012737 177776 001414      ;*REFORMAT SECTOR THAT WAS WRITTEN WITH BAD HEADER
2208 031532 012737 101360 001416      ;*****
2209 031540 012737 000062 001412      20$:
2210 031546 004737 035104      MOV      #FMT16,RMOFO  ;16 BIT MODE
2211
2212 031552 004737 032176      MOV      #-2,RMWCO     ;2 HEADER WORDS ONLY
      031556 154130      MOV      #BUFONE,RMBAO ;BUFFER ADDRESS
      .WORD 154130      MOV      #WH,RMCS10    ;WRITE HEAD AND DATA COMMAND
      .WORD 154130      JSR      PC,GENBUF     ;SET UP THE BUFFER
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
031560 000404      BR       21$          ;PREPARE DEVICE FOR TEST
031562 000240      NOP                      ;TASK DESCRIPTOR AS FOLLOWS:
031564 104000      EMT                      ;SELECT DEVICE & VERIFY DEVICE AVAILABLE
031566 000137 031572      EMT                      ;CLEAR CONTROLLER & SELECT DEVICE
2213 031572                      EMT                      ;VERIFY CONTROLLER CLEAR OPERATION
2214 031572 012737 000063 001412      JMP      21$          ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
2215 031600 012702 001553      MOV      #WH!GO,RMCS10 ;SET GO BIT
2216 031604 112722 000006      MOV      #PUTINX,R2    ;TABLE ADDRESS
2217 031610 112722 000034      MOV      #RMDA,(R2)+
2218 031614 112722 000032      MOV      #RMDC,(R2)+
2219 031620 112722 000004      MOV      #RMOF,(R2)+
2220 031624 112722 000002      MOV      #RMBA,(R2)+
2221 031630 112722 000000      MOV      #RMWC,(R2)+
2222 031634 112722 000200      MOV      #RMCS1,(R2)+
2223 031640 004737 036240      MOV      #200,(R2)+   ;TERMINATOR
      .WORD 036240      JSR      PC,PUT       ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
  
```

	031644	000404		BR	22\$:GO TO 22\$ IF NO ERROR
	031646	000240		NOP			:RETURN HERE IF ERROR
	031650	104000		EMT			:ERROR # DEFINED BY PUT SUBROUTINE
	031652	000137	031666	JMP	23\$:GO TO 23\$ IF ERROR
2224	031656						
2225	031656	000240		NOP			
2226	031660	004737	036602	JSR	PC, TIMEOUT		:WAIT FOR TIME OUT
2227	031664	000240		NOP			
2228	031666						

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

.SBTTL END OF SUB-PASS ROUTINE

: THIS IS THE END OF SUB-PASS ROUTINE. THIS ROUTINE IS USED TO
: TERMINATE THE OPERATION OF THE CURRENT DEVICE UNDER TEST AND
: SELECT THE NEXT DEVICE FOR TEST. IF THERE ARE NO MORE DEVICES
: TO TEST, EXIT IS MADE TO '\$EOP' ROUTINE. OTHERWISE, RETURN
: IS MADE TO 'SHUT' ROUTINE.

\$EOSP: SCOPE
NOP
MOV TSTQUE,RO ;GET POINTER TO TSTQUE
ADD #2,RO ;ADJUST POINTER TO NEXT DEVICE
MOV RO,TSTQUE ;SAVE POINTER TO TSTQUE
TST (RO) ;ANY MORE DEVICES FOR TEST ?
BEQ 1\$;BR IF NO
JMP SHUT ;JUMP TO 'SHUT' ROUTINE
1\$: MOV #TSTQUE+2,TSTQUE ;INITIALIZE POINTER TO FIRST DEVICE IN
;TEST QUE TABLE

.SBTTL END OF PASS ROUTINE

:*****
:*INCREMENT THE PASS NUMBER (\$PASS)
:*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY'
:*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
:*IF THERES A MONITOR GO TO IT
:*IF THERE ISN'T JUMP TO SHUT

031724
031724 000240
031726 005037 001116
031732 005037 001206
031736 005237 001230
031742 042737 100000 001230
031750 005327
031752 000001
031754 003066
031756 012737
031760 000001
031762 031752
031764 104401 031772
031770 000407

032010
032010 013746 001230

032014 104405
032016 005737 001126
032022 001431
032024 104401 032032
032030 000421

032074
032074 013746 001126

032100 104405
032102 005037 001126

\$EOP:
NOP
CLR \$TSTNM ;:ZERO THE TEST NUMBER
CLR \$TIMES ;:ZERO THE NUMBER OF ITERATIONS
INC \$PASS ;:INCREMENT THE PASS NUMBER
BIC #100000,\$PASS ;:DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;:LOOP?
\$EOPCT: .WORD 1
BGT \$DOAGN ;:YES
MOV (PC)+,@(PC)+ ;:RESTORE COUNTER
\$ENDCT: .WORD 1
TYPE ,65\$;:TYPE ASCIZ STRING
BR 64\$;:GET OVER THE ASCIZ
::65\$: .ASCIZ <1?><15>/END PASS #/
64\$: MOV \$PASS,-(SP) ;:SAVE \$PASS FOR TYPEOUT
;:TYPE PASS NUMBER
TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
TST \$ERTTL ;:SEE IF ANY ERRORS THIS PASS
BEQ \$GT42P ;:BR IF NO ERRORS TO REPORT
TYPE ,67\$;:TYPE ASCIZ STRING
BR 66\$;:GET OVER THE ASCIZ
::67\$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66\$: MOV \$ERTTL,-(SP) ;:SAVE \$ERTTL FOR TYPEOUT
;:TOTAL NUMBER OF ERRORS
TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
CLR \$ERTTL ;:CLEAR ERROR TOTAL

032106	104401	001217		\$GT42P: TYPE	,\$CRLF	::TYPE CARRIAGE RETURN, LINE FEED	
032112	013700	000042		\$GET42: MOV	@#42,RO	::GET MONITOR ADDRESS	
032116	001405			BEG	\$DOAGN	::BRANCH IF NO MONITOR	
032120	000005			RESET		::CLEAR THE WORLD	
032122	004710			\$ENDAD: JSR	PC,(RO)	::GO TO MONITOR	
032124	000240			NOP		::SAVE ROOM	
032126	000240			NOP		::FOR	
032130	000240			NOP		::ACT11	
032132				\$DOAGN:			
032132	000137			JMP	@(PC)+	::RETURN	
032134	032142			\$RTNAD: .WORD	SHUT		
032136	377	377	000	\$ENULL: .BYTE	-1,-1,0	::NULL CHARACTER STRING	
				.EVEN			
21							
22	032142	005737	001326	SHUT:	TST	CTLFG	:WAS CONTROL C FLAGGED ?
23	032146	001002		BNE	1\$:BR IF YES
24	032150	000137	007670	JMP	READY		:CONTINUE
25	032154	005737	000042	1\$:	TST	@#42	:ANY MONITOR PRESENT ?
26	032160	001002		BNE	2\$:BR IF YES
27	032162	000137	005432	JMP	START		:GO TO STAFF
28	032166	005037	001300	2\$:	CLR	\$DEVN	:FUDGE NO DRIVES IN MAP
29	032172	000137	031724	JMP	\$EOP		:RETURN TO \$EOP

.SBTTL TEST PREPARATION MODULE

: THIS MODULE PREPARES THE SUBSYSTEM FOR THE EXECUTION OF A TEST,
 : REPORTING AN ERROR TO THE USER IF AN ERROR IS DETECTED. THE USER
 : SPECIFIES TASKS TO BE PERFORMED, WHICH THE MODULE EXECUTES
 : USING SUBROUTINES.

:CALL:

```

    JSR    PC,TSTPRP          TASK/VERIFY DESCRIPTOR
    .WORD  NN'NNN            RETURN HERE IF NO ERROR
    BR     ??                 RETURN HERE IF ERROR
    NOP
    ERROR  ERROR DEFINED BY MODULE
    
```

:TASK/VERIFY DESCRIPTOR

```

    BIT 15 = 1                SELECT DEVICE AND VERIFY DEVICE IS AVAILABLE
    -----
    BIT 14 - 1                CLEAR CONTROLLER AND SELECT DEVICE
    BIT 13                    (RESERVED FOR DRIVE CLEAR)
    BIT 12 = 1                PACK ACKNOWLEDGE IF VOLUME NOT VALID
    -----
    BIT 11 = 1                RECALIBRATE IF POSITIONING IN PROGRESS OR SKI ERROR
    BIT 10 - 1                RECALIBRATE DRIVE
    BIT 9
    -----
    BIT 8
    BIT 7
    BIT 6 = 1                VERIFY CONTROLLER CLEAR OPERATION
    -----
    BIT 5                    (RESERVED FOR DRIVE CLEAR)
    BIT 4 - 1                VERIFY PACK ACKNOWLEDGE
    BIT 3 = 1                VERIFY RECALIBRATION
    -----
    BIT 2
    BIT 1
    BIT 0
    
```

TSTPRP:

:STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL

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

032176

```

032176 017637 000000 033146
032204 062716 000006
032210 105076 000000
032214 162716 000004
032220 004737 035704
032224 004737 035770
032230 000411
032232 000401
032234 000000
032236 062716 000004 1$:
032242 113776 032234 000000
032250 000137 033136
032254 013737 001400 033150 2$:
    
```

```

MOV    @ (SP),39$           ;STORE DESCRIPTOR
ADD    #6,(SP)              ;MOVE SP TO USERS ERROR CALL
CLRB   @ (SP)               ;CLEAR ERROR CALL
SUB    #4,(SP)              ;MOVE SP TO NO ERROR RETURN
JSR    PC,GETSTS            ;SETUP TO READ ALL REGISTERS
JSR    PC,GET                ;GET RMER2
BR     2$                   ;BR IF NO ERROR DETECTED
BR     1$                   ;GET OVER ERROR NUMBER
.WORD  0                    ;ERROR DEFINED BY GET SUBROUTINE
1$:   ADD    #4,(SP)         ;XFER ERROR TO USER AND
      MOVB   1$-2,@ (SP)    ;GET ERROR NUMBER.
      JMP    37$
2$:   MOV    RMER2I,40$      ;GET RMER2 AND SAVE FOR LATER
    
```

:*****

```

58 ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
59 032262 005737 033146 TST 39$ ;SELECT DEVICE??
60 032266 100014 BPL 4$ ;NO!!
61
62 032270 004737 043672 JSR PC,DEVSEL ;GO SELECT DEVICE
63 032274 000411 BR 4$ ;NO ERROR - CONTINUE
64 032276 000401 BR 3$
65 032300 000000 .WORD 0 ;ERROR NUMBER FROM DEVSEL
66 032302 062716 000004 3$: ADD #4,(SP) ;TRANSFER ERROR TO USER
67 032306 113776 032300 000000 MOVB 3$,2,@(SP)
68 032314 000137 033136 JMP 37$
69
70 ;*****
71 ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
72 032320 4$:
73 032320 032737 040000 033146 BIT #BIT14,39$ ;CLEAR CONTROLLER??
74 032326 001451 BEQ 13$ ;NO!!
75
76 032330 004737 045344 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
77 032334 000411 BR 7$ ;CONTINUE - NO ERROR
78 032336 000401 BR 6$
79 032340 000000 5$: .WORD 0 ;ERROR NUMBER FROM CNTCLR
80 032342 062716 000004 6$: ADD #4,(SP) ;TRANSFER ERROR TO USER
81 032346 113776 032340 000000 MOVB 5$,@ (SP)
82 032354 000137 033136 JMP 37$
83
84 ;*****
85 ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
86 032360 7$:
87 032360 032737 000100 033146 BIT #BIT6,39$ ;VERIFY??
88 032366 001431 BEQ 13$ ;NO!!
89
90 032370 004737 035770 JSR PC,GET ;GO GET STATUS
91 032374 000411 BR 10$ ;NO ERROR GETTING STATUS
92 032376 000401 BR 9$
93 032400 000000 8$: .WORD 0 ;ERROR FROM GETTING STATUS
94 032402 062716 000004 9$: ADD #4,(SP) ;TRANSFER ERROR TO USER
95 032406 113776 032400 000000 MOVB 8$,@ (SP)
96 032414 000137 033136 JMP 37$
97
98 032420 004737 045462 10$: JSR PC,CLRSTS ;GO VERIFY STATUS CLEAR
99 032424 000412 BR 13$ ;NO ERROR IN CLEAR
100 032426 000401 BR 12$
101 032430 000000 11$: .WORD 0 ;ERROR IN STATUS CLEAR
102 032432 005726 12$: TST (SP)+ ;STRIP RETURN ADDRESS TO
103 032434 062716 000004 ADD #4,(SP) ;SUBROUTINE AND TRANSFER
104 032440 113776 032430 000000 MOVB 11$,@ (SP) ;ERROR TO USER
105 032446 000137 033136 JMP 37$
106
107 ;*****
108 ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
109 ;NOT VALID
110 032452 13$:
111 032452 032737 010000 033146 BIT #BIT12,39$ ;PACK ACKNOWLEDGE??
112 032460 001501 BEQ 25$ ;NO!!
113
114 032462 004737 035770 JSR PC,GET

```

```

115 032466 000411 BR 16$ :NO ERROR GETTING RMDS
116 032470 000401 BR 15$
117 032472 000000 14$: .WORD 0
118 032474 062716 000004 15$: ADD #4,(SP) :TRANSFER ERROR TO USER
119 032500 113776 032472 000000 MOVB 14$,@ (SP)
120 032506 060137 033136 JMP 37$
121
122 032512 032737 000100 001350 16$: BIT #VV,RMDSI :IS VOLUME VALID??
123 032520 001061 BNE 25$ :YES!!
124
125 032522 012737 000023 001412 MOV #PAKACK!GO,RMCS10 :LOAD PACK ACK COMMAND
126 032530 112737 000000 001553 MOVB #RMCS1,PUTINX :SETUP REGISTER INDEX TABLE
127 032536 112737 000200 001554 MOVB #200,PUTINX+1
128 032544 004737 036240 JSR PC,PUT :GO WRITE COMMAND
129 032550 000410 BR 19$ :NO ERROR LOADING REGISTER
130 032552 000401 BR 18$
131 032554 000000 17$: .WORD 0 :ERROR FROM PUT SUB
132 032556 062716 000004 18$: ADD #4,(SP) :TRANSFER ERROR TO USER
133 032562 113776 032554 000000 MOVB 17$,@ (SP)
134 032570 000562 BR 37$
135
136 032572 004737 036602 19$: JSR PC,TIMOUT :WAIT FOR COMMAND TO COMPLETE
137
138 :*****
139 :VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
140 032576 032737 000020 033146 BIT #BIT4,39$ :VERIFY PACK ACKNOWLEDGE??
141 032604 001427 BEQ 25$ :NO!!
142
143 032606 004737 035770 JSR PC,GET :GO GET STATUS
144 032612 000410 BR 22$ :NO ERROR GETTING STATUS
145 032614 000401 BR 21$
146 032616 000000 20$: .WORD 0 :ERROR FROM GET SUB
147 032620 062716 000004 21$: ADD #4,(SP) :TRANSFER ERROR TO USER
148 032624 113776 032616 000000 MOVB 20$,@ (SP)
149 032632 000541 BR 37$
150
151 032634 004737 046342 22$: JSR PC,ACKSTS :GO CHECK ACKNOWLEDGE
152 032640 000411 BR 25$ :NO ERROR
153 032642 000401 BR 24$
154 032644 000000 23$: .WORD 0 :PACK ACKNOWLEDGE ERROR
155 032646 005726 24$: TST (SP)+ :STRIP RETURN TO SUB AND
156 032650 062716 000004 ADD #4,(SP) :TRANSFER ERROR TO USER
157 032654 113776 032644 000000 MOVB 23$,@ (SP)
158 032662 000525 BR 37$
159
160 :*****
161 :RECALIBRATE DRIVE IF BIT 11 IS SET IN TASK AND "SKI" IS SET
162 :OR "PIP" IS ACTIVE.
163 :RECALIBRATE DRIVE IF BIT 10 IS SET
164 032664 25$:
165 032664 032737 002000 033146 BIT #BIT10,39$ :RECALIBRATE DRIVE ?
166 032672 001027 BNE 29$ :YES.
167 032674 032737 004000 033146 BIT #BIT11,39$ :RECALIBRATE??
168 032702 001517 BEQ 38$ :NO!!
169
170 032704 004737 035770 JSR PC,GET :GO GET RMDS
171 032710 000410 BR 28$ :NO ERROR GETTING RMDS

```

```

172 032712 000401          BR      27$
173 032714 000000          .WORD  0          ;ERROR FROM GET SUB
174 032716 062716 000004  26$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
175 032722 113776 032714 000000  27$:   MOVB   26$,@ (SP)
176 032730 000502          BR      37$
177
178 032732 032737 040000 033150  28$:   BIT    #SKI,40$   ;WAS SKI SET ?
179 032740 001004          BNE    29$         ;YES, GO RECALIBRATE
180 032742 032737 020000 001350  BIT    #PIP,RMDSI  ;IS PIP ACTIVE??
181 032750 001474          BEQ    38$         ;NO..
182
183 032752 005037 001446          29$:   CLR    RMDCO      ;CLEAR CYLINDER ADDRESS
184 032756 005037 001420          CLR    RMDAO      ;CLEAR TRACK/SECTOR ADDRESS
185 032762 012737 000007 001412  MOV    #RECAL!GO,RMCS10 ;LOAD RECALIBRATE COMMAND
186 032770 112737 000034 001553  MOVB   #RMDL,PUTINX ;MAKE CYLINDER ADDRESS AND
187 032776 112737 000006 001554  MOVB   #RMDA,PUTINX+1 ;TRK/SEC ADDRESS LOOK LIKE SEEK TO ZERO
188 033004 112737 000000 001555  MOVB   #RMCS1,PUTINX+2 ;AND REGISTER INDEX
189 033012 112737 000200 001556  MOVB   #200,PUTINX+3 ;SET TERMINATOR
190 033020 004737 036240          JSR    PC,PUT      ;GO ISSUE RECALIBRATE
191 033024 000410          BR     31$        ;NO ERROR
192 033026 000401          BR     30$
193 033030 000000          .WORD  0          ;ERROR IN REGISTER TRANSFER
194 033032 062716 000004  30$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
195 033036 113776 033030 000000  MOVB   30$-2,@ (SP)
196 033044 000434          BR     37$
197
198 033046 004737 036602  31$:   JSR    PC,TIMOUT ;WAIT FOR COMPLETION
199
200
201
202 033052 032737 000010 033146  ;:*****
203 033060 001430          BIT    #BIT3,39$  ;VERIFY RECALIBRATE??
204
205 033062 004737 035770          BEQ    38$        ;NO!!
206 033066 000410          JSR    PC,GET     ;GO GET STATUS
207 033070 000401          BR     34$       ;NO ERROR GETTING STATUS
208 033072 000000          BR     33$
209 033074 062716 000004  32$:   .WORD  0          ;ERROR FROM GET
210 033100 113776 033072 000000  33$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
211 033106 000413          MOVB   32$,@ (SP)
212
213 033110 004737 047136  34$:   JSR    PC,RCLSTS ;GO CHECK RECALIBRATE
214 033114 000412          BR     38$       ;NO ERROR DURING RECALIBRATE
215 033116 000401          BR     36$
216 033120 000000          35$:   .WORD  0          ;ERROR DURING RECALIBRATE
217 033122 005726          36$:   TST    (SP)+     ;STRIP RETURN TO SUB AND
218 033124 062716 000004  37$:   ADD    #4,(SP)   ;TRANSFER ERROR TO USER
219 033130 113776 033120 000000  MOVB   35$,@ (SP)
220 033136 162716 000002  38$:   SUB    #2,(SP)   ;MOVE SP BACK BEFORE ERROR
221 033142 000240          38$:   NOP
222 033144 000207          RTS    PC        ;RETURN TO USER
223
224 033146 000000          39$:   .WORD  0          ;TASK/VERIFY DESCRIPTOR
225 033150 000000          40$:   .WORD  0          ;CONTAINS RMER2
    
```

.SBTTL BAD SECTOR MODULE

:THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
 :GENERATOR SUBROUTINE, AND PRESERVES THE 'PUT BUFFER' SO THAT THE
 :BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
 :OPERATION.

:THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
 :SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS 'ASNDA' AND 'ASNDL'
 :SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.

:THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS:

- (1) RECOVER THE BAD SECTOR FILES AND
- (2) APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
 THE BAD SECTOR FILES OR ASSIGN A NEW SECTOR IF THE ONE
 ELECTED IS NOT AVAILABLE FOR USE.

:INFORMATION REQUIRED BY THE MODULE INCLUDES:

- (1) .RMDCO - THE DESIRED CYLINDER,
- (2) .RMDAO - THE TRACK AND SECTOR ADDRESS,
- (3) .RMWCO - THE WORD COUNT,
- (4) .RMCS10 - THE COMMAND,
- (5) .RMOFO - THE FORMAT MODE

:CALL:

```

JSR    PC,BADSCT    ;CALL SUBROUTINE
BR     ???          ;RETURN HERE IF NO ERROR
TYPE   ,MESSAGE     ;RETURN HERE IF THE BAD SECTOR FILE
                     ;CANNOT BE RECOVERED.
ERROR  N            ;THE EMT OFFSET NUMBER 'N' IS DEFINED
                     ;BY BAD SECTOR MODULE.
    
```

BADSCT:

```

ADD    #6,(SP)      ;CLEAR ERROR NUMBER IN USER'S
CLR    @2(SP)       ;ERROR CALL.
SUB    #6,(SP)
    
```

:TEST 'MEDIA ENABLE' TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
 :HAVE BEEN RECOVERED.

```

TST    MEDENB      ;HAS BAD SECTOR FILES BEEN RECOVERED ?
BEQ    1$          ;BR IF NO
JMP    54$         ;YES, BAD SECTOR FILE IS AVAILABLE
    
```

:RECOVER THE MANUFACTURES / USERS BAD SECTOR FILE FROM CYLINDER = 822.
 :TRACK - LAST TRACK (RM02/3 - 4 AND RM05 18.). ALSO, SAVE THE USER'S
 :PUT BUFFER

```

1$:
MOV    R0,-(SP)    ;;PUSH R0 ON STACK
CLR    R0          ;START WITH RMCS1
2$:
MOV    PUTBUF(R0),BUFFER(R0)
ADD    #2,R0       ;ADVANCE TO NEXT BUFFER POSITION
CMP    #46,R0      ;END OF BUFFER
BHS    2$         ;NO !!
    
```

:SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
 :SETUP PARAMETERS TO READ SKIP SECTOR FILE FIRST (3 SECTORS)

```

MOV    #3,68$     ;RETRY COUNT
    
```

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

```

033152
033152 062716 000006
033156 105076 000000
033162 162716 000006

033166 005737 001512
033172 001402
033174 000137 034452

033200
033200 010046
033202 005000
033204 016060 001412 101360
033212 062700 000002
033216 022700 000046
033222 103370

033224 012737 000003 035070
    
```

```

57 033232 012737 001466 001446      MOV      #822.,RMDCO      ;DESIRED CYLINDER = 822.
58 033240 013737 001334 001420      MOV      LSTRK,RMDAO     ;STARTING LAST TRACK, SECTOR = 0
59 033246 012737 177376 001414      MOV      #-258.,RMWCO    ;2 + 256. WORDS (2'S COMP)
60 033254 012737 010000 001444      MOV      #FMT16,RMOFO    ;16 BIT FORMAT
61 033262 012737 103370 001416      MOV      #MFGFIL,RMBAO   ;POINT TO MANUFACTURES FILE BUFFER
62
63 033270 012700 001553                MOV      #PUTINX,RO      ;RO POINTS TO REGISTER INDEX TABLE
64 033274 112720 000006      MOVVB   #RMDA,(RO)+
65 033300 112720 000034      MOVVB   #RMDC,(RO)+
66 033304 112720 000002      MOVVB   #RMWC,(RO)+
67 033310 112720 000032      MOVVB   #RMOF,(RO)+
68 033314 112720 000004      MOVVB   #RMBA,(RO)+
69 033320 112720 000000      MOVVB   #RMCS1,(RO)+
70 033324 112720 000200      MOVVB   #200,(RO)+
71 033330 012600                MOV      (SP)+,RO       ;;POP STACK INTO RO
72
73                                ;SET GET INDEX TABLE FOR READING STATUS
74 033332                3$:
75 033332 004737 035704      JSR      PC,GETSTS      ;SETUP GET INDEX REGISTER FOR STATUS
76 033336 004737 035770      JSR      PC,GET        ;GET REGISTERS
77 033342 000411                BR      5$              ;BR IF NO ERROR
78 033344 000401                BR      4$              ;JUMP OVER ERROR NUMBER
79 033346 000000                .WORD   0              ;ERROR DEFINED BY GET SUB
80 033350 062716 000006      4$:      ADD      #6,(SP)      ;XFER ERROR TO USER AND
81 033354 113776 033346 000000      MOVVB   4$-2,@(SP)    ;GET ERROR NUMBER.
82 033362 000137 034156      JMP      42$
83
84 033366 013737 004400 035102 5$:      MOV      RMER2I,73$    ;GET RMER2 AND SAVE FOR LATER
85
86                                ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
87 033374 012737 000011 001412      MOV      #DRVCLR.GO,RMCS10 ;LOAD COMMAND IN PUT BUFFER
88 033402 004737 036240      JSR      PC,PUT        ;OUTPUT COMMAND
89 033406 000411                BR      8$              ;RETURN HERE IF NO ERROR
90 033410 000401                BR      7$              ;GET AROUND ERROR #
91 033412 000000                .WORD   0              ;ERROR # GOES HERE
92 033414 062716 000006      6$:      .WORD   0
93 033420 113776 033412 000000      7$:      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
94 033426 000137 034156      MOVVB   6$,@(SP)     ;MOVE ERROR NUMBER TO USER
95                                JMP      42$
96 033432 004737 036602      8$:      JSR      PC,TIMOUT     ;WAIT FOR COMPLETION
97 033436 004737 035770      JSR      PC,GET        ;GO GET STATUS
98 033442 000411                BR      11$             ;RETURN HERE IF NO ERROR
99 033444 000401                BR      10$             ;GET AROUND ERROR #
100 033446 000000                .WORD   0              ;ERROR # GOES HERE
101 033450 062716 000006      9$:      .WORD   0
102 033454 113776 033446 000000      10$:     ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
103 033462 000137 034156      MOVVB   9$,@(SP)    ;MOVE ERROR # TO USERS ERROR CALL
104                                JMP      42$
105 033466 004737 050700      11$:     JSR      PC,DRVSTS     ;GO VERIFY DRIVE CLEAR COMMAND
106 033472 000412                BR      14$             ;RETURN HERE IF NO ERROR
107 033474 000401                BR      13$             ;GET AROUND ERROR
108 033476 000000                .WORD   0              ;ERROR # GOES HERE
109 033500 005726      12$:     .WORD   0
110 033502 062716 000006      13$:     TST      (SP)+        ;STRIP RETURN TO SUBROUTINE
111 033506 113776 033476 000000      ADD      #6,(SP)      ;MOVE SP TO USERS ERROR CALL
112 033514 000137 034156      MOVVB   12$,@(SP)   ;MOVE ERROR # TO USER CALL
113                                JMP      42$

```



```

114
115 033520
116 033520 032737 000100 001350
117 033526 001052
118
119 033530 012737 000023 001412
120 033536 004737 036240
121 033542 000411
122 033544 000401
123 033546 000000
124 033550 062716 000006
125 033554 113776 033546 000000
126 033562 000137 034156
127
128 033566 004737 036602
129 033572 004737 035770
130 033576 000411
131 033600 000401
132 033602 000000
133 033604 062716 000006
134 033610 113776 033602 000000
135 033616 000137 034156
136
137 033622 004737 046342
138 033626 000412
139 033630 000401
140 033632 000000
141 033634 005726
142 033636 062716 000006
143 033642 113776 033632 000000
144 033650 000137 034156
145
146
147 033654
148 033654 032737 040000 035102
149 033662 001004
150 033664 032737 020000 001350
151 033672 001452
152
153 033674 012737 000007 001412
154 033702 004737 036240
155 033706 000411
156 033710 000401
157 033712 000000
158 033714 062716 000006
159 033720 113776 033712 000000
160 033726 000137 034156
161
162 033732 004737 036602
163 033736 004737 035770
164 033742 000411
165 033744 000401
166 033746 000000
167 033750 062716 000006
168 033754 113776 033746 000000
169 033762 000137 034156
170
    ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
14$:
    BIT #VV,RMDSI ;IS VV RESET ??
    BNE 23$ ;NO !!
    MOV #PACK.GO,RMCS10 ;LOAD COMMAND
    JSR PC,PUT ;GO PUT COMMAND TO DRIVE
    BR 17$ ;RETURN HERE IF NO OUTPUT ERROR
    BR 16$ ;GET AROUND ERROR #
    .WORD 0 ;ERROR # GOES HERE
15$:
16$:
    ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
    MOVB 15$,@ (SP) ;MOVE ERROR # TO ERROR CALL
    JMP 42$
17$:
    JSR PC,TIMOUT ;WAIT FOR COMPLETION
    JSR PC,GET ;GO GET STATUS FOR PACK ACK
    BR 20$ ;RETURN HERE IF NO ERROR
    BR 19$ ;GET AROUND ERROR #
    .WORD 0 ;ERROR # GOES HERE
18$:
19$:
    ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
    MOVB 18$,@ (SP) ;MOVE ERROR # TO CALL
    JMP 42$
20$:
    JSR PC,ACKSTS ;GO VERIFY ACKNOWLEDGE STATUS
    BR 23$ ;RETURN HERE IF NO ERROR
    BR 22$ ;GET AROUND ERROR #
    .WORD 0 ;ERROR # GOES HERE
21$:
22$:
    TST (SP)+ ;STRIP RETURN TO SUBROUTINE
    ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
    MOVB 21$,@ (SP) ;MOVE ERROR # TO USERS ERROR CALL
    JMP 42$
    ;RECALIBRATE THE DRIVE IF 'SKI' OR 'PIP' IS SET
23$:
    BIT #SKI,73$ ;WAS SKI SET ?
    BNE 24$ ;YES, GO RECALIBRATE
    BIT #PIP,RMDSI ;IS PIP SET ??
    BEQ 32$ ;NO !!
    MOV #RECAL!GO,RMCS10 ;LOAD RECALIBRATE COMMAND
    JSR PC,PUT ;PUT THE RECAL COMMAND
    BR 26$ ;RETURN HERE IF NO ERROR
    BR 25$ ;GET AROUND ERROR #
    .WORD 0 ;ERROR # GOES HERE
24$:
25$:
    ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
    MOVB 25$-2,@ (SP) ;MOVE ERROR # TO USERS CALL
    JMP 42$
26$:
    JSR PC,TIMOUT ;WAIT FOR RECALIBRATE TO COMPLETE
    JSR PC,GET ;GO GET RECAL STATUS
    BR 29$ ;RETURN HERE IF NO ERROR
    BR 28$ ;GET AROUND ERROR #
    .WORD 0 ;ERROR # GOES HERE
27$:
28$:
    ADD #6,(SP) ;MOVE SP TO USERS ERROR CALL
    MOVB 27$,@ (SP) ;MOVE ERROR TO USERS CALL
    JMP 42$
    
```

```

171 033766 004737 047136      29$: JSR PC,RCLSTS      ;GO VERIFY RECALIBRATE STATUS
172 033772 000412              BR 32$                ;RETURN HERE IF NO ERROR
173 033774 000401              BR 31$                ;GET AROUND ERROR #
174 033776 000000              .WORD 0                ;ERROR # GOES HERE
175 034000 005726              30$: TST (SP)+          ;STRIP RETURN TO SUBROUTINE
176 034002 062716 000006      31$: ADD #6,(SP)          ;MOVE SP TO USERS ERROR CALL
177 034006 113776 033776 000000 MOVB 30$,a(SP)         ;MOVE ERROR # TO USERS CALL
178 034014 000137 034156      JMP 42$
179
180 ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
181 034020      32$: MOV #RH.GO,RMCS10    ;LOAD READ HEADER AND DATA COMMAND
182 034020 012737 000073 001412 JSR PC,PUT            ;PUT COMMAND
183 034026 004737 036240      BR 35$                ;RETURN HERE IF NO ERROR
184 034032 000411              BR 34$                ;GET AROUND ERROR #
185 034034 000401              .WORD 0                ;ERROR # GOES HERE
186 034036 000000      33$: ADD #6,(SP)          ;MOVE SP TO USERS ERROR CALL
187 034040 062716 000006      34$: MOVB 33$,a(SP)         ;MOVE ERROR # TO USERS ERROR CALL
188 034044 113776 034036 000000 JMP 42$
189 034052 000137 034156
190
191 034056 004737 036602      35$: JSR PC,TIMOUT      ;WAIT FOR READ OPERATION TO COMPLETE
192 034062 004737 035770      JSR PC,GET            ;GO GET STATUS FOR READ OPERATION
193 034066 000411              BR 38$                ;RETURN HERE IF NO ERROR
194 034070 000401              BR 37$                ;GET AROUND ERROR #
195 034072 000000              .WORD 0                ;ERROR # GOES HERE
196 034074 062716 000006      36$: ADD #6,(SP)          ;MOVE SP TO USERS ERROR CALL
197 034100 113776 034072 000000 MOVB 36$,a(SP)         ;MOVE ERROR # TO CALL
198 034106 000137 034156      JMP 42$
199
200 034112 004737 051502      38$: JSR PC,DTASTS      ;GO VERIFY RESULTS OF READ OPERATION
201 034116 000412              BR 41$                ;RETURN HERE IF NO ERROR
202 034120 000401              BR 40$                ;GET AROUND ERROR #
203 034122 000000              .WORD 0                ;ERROR # GOES HERE
204 034124 005726              39$: TST (SP)+          ;STRIP RETURN ADDRESS TO SUBROUTINE
205 034126 062716 000006      40$: ADD #6,(SP)          ;MOVE SP TO USERS ERROR CALL
206 034132 113776 034122 000000 MOVB 39$,a(SP)         ;MOVE ERROR # TO USERS CALL
207 034140 000137 034156      JMP 42$
208
209 034144 032737 040000 001336 41$: BIT #TRE,RMCS11    ;ANY CONTROLLER ERRORS ?
210 034152 001001              BNE 42$                ;BR IF YES
211 034154 000446              BR 48$                ;NO ERRORS DETECTED
212
213 ;*****
214 ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
215 ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
216
217 034156 005337 035070      42$: DEC 68$            ;YES, DECREMENT RETRY COUNT AND
218 034162 100026              BPL 45$                ;RETRY IF COUNT NOT NEGATIVE.
219
220 ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
221
222 034164 032737 100720 001352 43$: BIT #DCK.HCRC!HCE!FER!ECH,RMER1I ;ANY MEDIA RELATED ERRORS ?
223 034172 001004              BNE 44$                ;YES, GO TRY NEXT AVAILABLE SECTOR
224
225 034174 032737 100000 001400 BIT #BSE,RMER2I      ;ANY MEDIA RELATED ERRORS ?
226 034202 001422              BEQ 46$                ;NO, EXIT AND REPORT ERROR ON RETURN
227

```

```

229                                     ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
230                                     ;DUE TO THE MEDIA. SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
231                                     ;ANOTHER AREA ON THE LAST TRACK
232 034204 062737 000002 001420 44$:  ADD    #2,RMDAO      ;ADVANCE SECTOR ADDRESS BY 2
233 034212 122737 000012 001420      CMPB   #10.,RMDAO    ;QUIT IF ALL MFG SECTORS HAVE BEEN
234 034220 001413      BEQ    46$          ;TRIED.
235 034222 122737 000040 001420      CMPB   #32.,RMDAO   ;QUIT IF ALL USER SECTORS HAVE BEEN
236 034230 001407      BEQ    46$          ;TRIED.
237
238 034232 012737 000003 035070      MOV    #3,68$      ;REINSTATE RETRY COUNT FOR THIS SECTOR
239 034240 162716 000006      45$:  SUB    #6,(SP)   ;MOVE SP BACK TO NO ERROR
240 034244 000137 033332      JMP    3$          ;RETRY THE READ OPERATION
241
242                                     ;THE BAD SECTOR FILE CANNOT BE READ
243
244 034250 000240      46$:  NOP
245 034252 032777 020000 144674      BIT    #SW13,@SWR  ;INHIBIT MESSAGE ?
246 034260 001002      BNE   47$          ;YES
247 034262 162716 000004      SUB    #4,(SP)    ;MOVE SP TO ERROR RETURN
248 034266 000137 035064      47$:  JMP    67$          ;GO TO MODULE EXIT
249
250                                     ;THE SECTOR WAS RECOVERED WITHOUT ERROR - READ THE USER FILE IF
251                                     ;THIS IS THE MGF FILE OR ELSE DONE.
252
253 034272 022737 104376 001416 48$:  CMP    #USRFIL,RMBAO ;WAS THE USER FILE READ ??
254 034300 001446      BEQ    52$          ;YES - READ IS COMPLETE
255 034302 112737 000012 001420      MOVB   #10.,RMDAO  ;READ THE USER FILE LAST TRACK, SECTOR 10.
256 034310 012737 104376 001416      MOV    #USRFIL,RMBAO ;POINT TO USERS FILE BUFFER
257
258 034316 012737 000003 035070      MOV    #3,68$      ;RELOAD THE RETRY COUNT FOR THIS SECTOR
259 034324 000137 033332      JMP    3$          ;GO READ THE USER FILE
260
261                                     ;DUMMY THE BAD SECTOR FILES
262 034330      49$:
263 034330 010046      MOV    R0,-(SP)    ;:PUSH R0 ON STACK
264 034332 010146      MOV    R1,-(SP)    ;:PUSH R1 ON STACK
265 034334 012701 000374      MOV    #252.,R1   ;R1 = NUMBER OF ENTRIES IN FILES
266 034340 012700 000014      MOV    #14,R0     ;R0 = ADDRESS INDEX TO FILE STORAGE
267 034344 012760 177777 103370 50$:  MOV    #-1,MFGFIL(R0) ;ENTER ALL ONES IN MFG FILE
268 034352 012760 177777 104376      MOV    #-1,USRFIL(R0) ;ENTER ALL ONES IN USER FILE
269 034360 005720      TST   (R0)+       ;ADVANCE ADDRESS
270 034362 005301      DEC   R1          ;DECREMENT COUNT
271 034364 001367      BNE   50$        ;CONTINUE IF NOT DONE
272
273 034366 012701 000006      MOV    #6.,R1     ;CLEAR HEADER, CLEAR ID & SERIAL NUMBERS
274 034372 005000      CLR   R0
275 034374 005060 103370 51$:  CLR   MFGFIL(R0)
276 034400 005060 104376      CLR   USRFIL(R0)
277 034404 005720      TST   (R0)+       ;ADVANCE ADDRESS
278 034406 005301      DEC   R1
279 034410 001371      BNE   51$
280 034412 012601      MOV   (SP)+,R1   ;:POP STACK INTO R1
281 034414 012600      MOV   (SP)+,R0   ;:POP STACK INTO R0
                                     ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
    
```

```

282 034416 010046          MOV    R0,-(SP)          ;;PUSH R0 ON STACK
283 034420 005000          CLR    R0                ;;R0 IS REGISTER INDEX
283 034422 012737 177777 001512  MOV    #-1,MEDENB
284 034430 016060 101360 001412 53$:  MOV    BUFFER(R0),PUTBUF(R0)
285 034436 062700 000012  ADD    #2,R0              ;;ADVANCE R0
286 034442 022700 000046  CMP    #46,R0             ;;DONE ??
287 034446 103370          BHIS   53$
288 034450 012600          MOV    (SP)+,R0          ;;POP STACK INTO R0
289
290
291
292
293
294
295
296
297 034452          ;;*****
          ;;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
          ;;AND NOT IN THE USERS BAD SECTOR FILE. ASSIGN A NEW SECTOR IF THE
          ;;SECTOR IS IN ANY OF THE FILES.
          ;;*****
          ;;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
          54$:
          MOV    R0,-(SP)          ;;PUSH R0 ON STACK
          MOV    R1,-(SP)          ;;PUSH R1 ON STACK
          MOV    R2,-(SP)          ;;PUSH R2 ON STACK
298 034460 013737 001446 001514  MOV    RMDCO,ASNDC        ;;LOAD REQUESTED CYLINDER, TRACK,
299 034466 013737 001420 001516  MOV    RMDAO,ASNDA        ;;AND SECTOR ADDRESS IN ASSIGNED STORAGE
300 034474 005002          CLR    R2                ;;R2 = NUMBER OF SECTORS
301 034476 013700 001414  MOV    RMWCO,R0           ;;R0 = WORD COUNT
302 034502 005400          NEG    R0                ;;MAKE NUMBER POSITIVE
303 034504 012701 000400  MOV    #256.,R1           ;;R1 = NUMBER OF WORDS PER SECTOR
304 034510 032737 000002 001412  BIT    #BIT1,RMCS10        ;;IS THIS A HEADER AND DATA COMMAND ??
305 034516 001402          BEQ    55$               ;;NO !!
306 034520 012701 000402  MOV    #258.,R1           ;;CHANGE WORDS PER SECTOR
307 034524 020100 55$:  CMP    R1,R0              ;;IS THERE A FULL SECTOR ??
308 034526 101404          BLOS   56$               ;;YES !!
309 034530 005700          TST    R0                ;;IS R0 ZERO ??
310 034532 001405          BEQ    57$               ;;YES !!
311 034534 005202          INC    R2                ;;INCREMENT FOR PARTIAL SECTOR
312 034536 000403          BR    57$
313 034540 160100 56$:  SUB    R1,R0              ;;SUBTRACT ONE SECTOR FROM WORD COUNT
314 034542 005202          INC    R2                ;;INCREMENT SECTOR COUNT
315 034544 000767          BR    55$
316 034546 010237 035070 57$:  MOV    R2,68$            ;;SAVE SECTOR COUNT
317
318
319
320
321
322 034552 012737 103404 035100  MOV    #MFGFIL+14,72$     ;;THE STARTING ADDRESS OF MFG FILE
323
324 034560 004737 034602          JSR    PC,58$             ;;TO BASE ADDRESS STORAGE.
325 034564 012737 104412 035100  MOV    #USRFIL+14,72$     ;;GO SEARCH FILE
326
327 034572 004737 034602          JSR    PC,58$             ;;LOAD STARTING ADDRESS OF USR FILE
328 034576 000137 035042          JMP    66$               ;;TO BASE ADDRESS STORAGE.
329
330 034602 013737 001514 035074 58$:  MOV    ASNDC,70$          ;;LOAD COMPARING CYLINDER ADDRESS
331 034610 013737 001516 035076  MOV    ASNDA,71$          ;;LOAD COMPARING TRACK, SECTOR ADDRESS
332 034616 013737 035070 035072  MOV    68$,69$           ;;LOAD NUMBER OF SECTORS TO CONFIRM
333
334
          ;;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
    
```

```

335      ;CYLINDER, TRACK AND SECTOR ADDRESS
336 034624
337 034624 013700 035100
338 034630 022710 177777
339 034634 001446
340 034636 021037 035074
341 034642 001010
342
343      ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
344      ;THE COMPARING TRACK, AND SECTOR.
345 034644
346 034644 126037 000003 035077
347 034652 001004
348
349 034654 126037 000002 035076
350 034662 001402
351
352 034664 022020
353 034666 000760
354
355      ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
356      ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
357 034670
358 034670 105237 001516
359 034674 122737 000037 001516
360 034702 103337
361 034704 105037 001516
362 034710 105237 001517
363 034714 123737 001335 001517
364 034722 103327
365 034724 005037 001516
366 034730 005237 001514
367 034734 022737 001466 001514
368 034742 103317
369 034744 005037 001514
370 034750 000714
371
372      ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
373      ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
374      ;IS NOT ZERO.
375 034752
376 034752 005337 035072
377 034756 001442
378
379 034760 105237 035076
380 034764 122737 000037 035076
381 034772 103022
382 034774 105037 035076
383 035000 105237 035077
384 035004 123737 001335 035077
385 035012 103012
386 035014 005037 035076
387 035020 005237 035074
388 035024 022737 001466 035074
389 035032 103002
390 035034 005037 035074
391

```

```

59$:      MOV      72$,RO      ;LOAD THE BASE ADDRESS IN RO
60$:      CMP      #-1,(RO)    ;IS THIS FILE TERMINATOR ?
          BEQ      64$        ;BR IF YES
          CMP      (RO),70$    ;DOES TABLE ENTRY = COMPARING CYLINDER ?
          BNE      62$        ;BR IF NO

61$:      CMPB     3(RO),71$+1  ;DOES TABLE ENTRY = COMPARING TRACK ?
          BNE      62$        ;BR IF NO

62$:      CMP      (RO)+,(RO)+ ;NO, ADJUST CYLINDER POINTER IN BAD FILE
          BR       60$        ;AND CONTINUE SEARCH.

63$:      INCB     ASNDA        ;INCREMENT SECTOR
          CMPB     #31.,ASNDA   ;SECTOR OK ??
          BHIS     58$         ;YES !!
          CLRB     ASNDA        ;CLEAR SECTOR AND ADVANCE TRACK
          INCB     ASNDA+1
          CMPB     LSTRK+1,ASNDA+1 ;TRACK OK ?
          BHIS     58$         ;YES !!
          CLR      ASNDA        ;CLEAR TRACK AND SECTOR
          INC      ASNDC        ;INCREMENT CYLINDER
          CMP      #822.,ASNDC  ;CYLINDER OK ??
          BHIS     58$         ;YES !!
          CLR      ASNDC        ;START AT CYLINDER 0
          BR       58$         ;SEARCH NEXT SECTOR

64$:      DEC      69$         ;DECREMENT NUMBER OF SECTORS TO COMPARE
          BEQ      67$         ;DONE IF ZERO

          INCB     71$         ;INCREMENT THE COMPARING SECTOR
          CMPB     #31.,71$     ;SECTOR OK ??
          BHIS     65$         ;YES !!
          CLRB     71$         ;CLEAR SECTOR
          INCB     71$+1        ;INCREMENT TRACK
          CMPB     !STRK+1,71$+1 ;TRACK OK ??
          BHIS     65$         ;YES !!
          CLR      71$         ;CLEAR SECTOR TRACK
          INC      70$         ;INCREMENT CYLINDER
          CMP      #822.,70$    ;CYLINDER OK ??
          BHIS     65$         ;YES !!
          CLR      70$         ;START AT CYLINDER 0

```

392 035040 000671
 393
 394
 395 035042
 396 035042 013737 001514 001446
 397 035050 013737 001516 001420
 398 035056 012602
 035060 012601
 035062 012600
 399
 400 035064 000240
 401 035066 000207
 402
 403
 404
 405 035070 000000
 406 035072 000000
 407 035074 000000
 408 035076 000000
 409 035100 000000
 410 035102 000000

65\$: BR 59\$;SEARCH NEXT SECTOR

;ASSIGN THE SECTOR AND RETURN TO USER

66\$:
 MOV ASNDC,RMDCO ;LOAD CYLINDER
 MOV ASNDA,RMDAO ;LOAD TRACK AND SECTOR
 MOV (SP)+,R2 ;POP STACK INTO R2
 MOV (SP)+,R1 ;POP STACK INTO R1
 MOV (SP)+,R0 ;POP STACK INTO R0

67\$: NOP
RTS PC

;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE

68\$: .WORD 0 ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
 69\$: .WORD 0 ;NUMBER OF SECTORS TO COMPARE
 70\$: .WORD 0 ;COMPARING CYLINDER
 71\$: .WORD 0 ;COMPARING TRACK AND SECTOR
 72\$: .WORD 0 ;BASE ADDRESS OF BAD SECTOR FILE BEING SEARCHED
 73\$: .WORD 0 ;CONTAINS RMER2

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

.SBTTL BUFFER GENERATOR SUBROUTINE

: THIS SUBROUTINE GENERATES A DATA BUFFER FOR WRITE COMMANDS. THE
 : BUFFER STARTS AT RMBA AND IS RMWC WORDS LONG. THE BUFFER
 : CONTAINS A REPETITIVE DATA PATTERN CONSISTING OF \$TMP1 WORDS
 : FROM THE DATA PATTERN TABLE, BEGINNING AT ADDRESS \$TMP0.
 : HEADER INFORMATION FOR THE BUFFER IS EXTRACTED FROM RMDC,
 : RMDA AND RMOF.

: R0 = ADDRESS OF DATA BUFFER
 : R1 = LENGTH OF DATA BUFFER
 : R2 = ADDRESS OF DATA PATTERN
 : R3 = LENGTH OF DATA PATTERN
 : R4 = SECTOR COUNT

: CALL:
 : (1) JSR PC,GENBUF
 : (2) ?? RETURN HERE

GENBUF:

035104	010046				MOV	R0,-(SP)	::PUSH R0 ON STACK
035106	010146				MOV	R1,-(SP)	::PUSH R1 ON STACK
035110	010246				MOV	R2,-(SP)	::PUSH R2 ON STACK
035112	010346				MOV	R3,-(SP)	::PUSH R3 ON STACK
035114	010446				MOV	R4,-(SP)	::PUSH R4 ON STACK
035116	013700	001416			MOV	RMBA0,R0	:LOAD DATA BUFFER ADDRESS
035122	013701	001414			MOV	RMWC0,R1	:LOAD WORD COUNT
035126	013737	001446	035336		MOV	RMDC0,60\$:LOAD STARTING CYLINDER ADDRESS
035134	013737	001420	035340		MOV	RMDA0,65\$:LOAD STARTING TRACK,SECTOR ADDRESS
035142	032737	000002	001412	10\$:	BIT	#BIT1,RMCS10	:WRITE HEADER & DATA??
035150	001445				BEQ	25\$:NO!!
035152	013710	035336			MOV	60\$,(R0)	:WRITE HEADER WORD #1
035156	052710	140000			BIS	#MSE!USE,(R0)	:SET BAD SECTOR FLAGS FOR GOOD SECTOR
035162	012702	000035			MOV	#29.,R2	:R2 = MAXIMUM SECTOR ADDRESS (29.)
035166	032737	010000	001444		BIT	#FMT16,RMOFO	:18 BIT FORMAT??
035174	001404				BEQ	15\$:YES !!
035176	052710	010000			BIS	#FMT16,(R0)	:SET 16 FORMAT BIT IN HEADER
035202	012702	000037			MOV	#31.,R2	:CHANGE MAXIMUM SECTOR ADDRESS (31.)
035206	005201			15\$:	INC	R1	:INCREMENT WORD COUNT
035210	001443				BEQ	50\$:EXIT IF DONE
035212	005720				TST	(R0)+	:MOVE R0 TO HEADER WORD #2
035214	013720	035340			MOV	65\$,(R0)+	:WRITE HEADER WORD #2
035220	005201				INC	R1	:INCREMENT WORD COUNT AND
035222	001436				BEQ	50\$:EXIT IF DONE
035224	012703	035740			MOV	#65\$,R3	:ADVANCE SECTOR ADDRESS
035230	105213				INCB	(R3)	
035232	120213				CMPB	R2,(R3)	:SECTOR OVERFLOW ??
035234	103013				BHIS	25\$:NO !
035236	105013				CLRB	(R3)	:YES - CLEAR SECTOR ADDRESS
035240	105263	000001			INCB	1(R3)	:ADVANCE TRACK ADDRESS
035244	123763	001335	000001		CMPB	LSTRK+1,(R3)	:TRACK OVERFLOW ??
035252	103004				BHIS	25\$:NO !!
035254	105063	000001			CLRB	1(R3)	:YES - CLEAR TRACK ADDRESS

53	035260	105237	035336		INCB	60\$:ADVANCE CYLINDER ADDRESS
54	035264	012704	000400	25\$:	MOV	#256,R4		:LOAD SECTOR DATA COUNT
55	035270	013702	001174	30\$:	MOV	\$TMP0,R2		:LOAD PATTERN ADDRESS
56	035274	013703	001176		MOV	\$TMP1,R3		:LOAD PATTERN COUNT
57	035300	012220		40\$:	MOV	(R2)+,(R0)+		:WRITE DATA PATTERN
58	035302	005201			INC	R1		:INCREMENT WORD COUNT AND
59	035304	001405			BEQ	50\$:EXIT IF DONE
60	035306	005304			DEC	R4		:DECREMENT SECTOR COUNT
61	035310	001714			BEQ	10\$:START NEXT SECTOR IF 0
62	035312	005303			DEC	R3		:DECREMENT PATTERN COUNT
63	035314	001765			BEQ	30\$:RESTART PATTERN IF 0
64	035316	000770			BR	40\$:CONTINUE DATA PATTERN
65	035320			50\$:				
	035320	012604			MOV	(SP)+,R4		::POP STACK INTO R4
	035322	012603			MOV	(SP)+,R3		::POP STACK INTO R3
	035324	012602			MOV	(SP)+,R2		::POP STACK INTO R2
	035326	012601			MOV	(SP)+,R1		::POP STACK INTO R1
	035330	012600			MOV	(SP)+,R0		::POP STACK INTO R0
66	035332	000240			NOP			
67	035334	000207			RTS	PC		
68								
69	035336	000000		60\$:	.WORD			:CYLINDER ADDRESS STORAGE
70	035340	000000		65\$:	.WORD			:TRACK, SECTOR ADDRESS STORAGE


```

1      .SBTTL COMPARE BUFFER SUBROUTINE
2
3      ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
4      ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
5      ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ. ERRORS IN BUFFER
6      ;COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.
7
8      ;CALL:
9      ;(1) JSR PC,CMPBUF
10     ;(2) .WORD WRITE BUFFER ADDRESS
11     ;(3) .WORD READ BUFFER ADDRESS
12     ;(4) BR ?? RETURN HERE IF NO ERROR
13     ;(5) NOP RETURN HERE IF ERROR
14     ;(6) ERROR ERROR DEFINED BY SUBROUTINE
15     ;(7) ???
16
17     CMPBUF:
18     035342 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
19     035344 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
20     035346 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
21     035350 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
22     035352 005037 CLR 150$ ;;CLEAR CORRECTION FLAG
23
24     ;DETERMINE IF DATA SHOULD BE CORRECTED
25     035356 033737 004000 001370 BIT ECI,RMOFI ;WAS ECC CORRECTION ALLOWED ??
26     035364 001063 BNE 80$ ;NO !!
27     035366 032737 100000 00352 BIT #DCK,RMER1I ;WAS THERE A DATA CHECK ??
28     035374 001457 BEQ 80$ ;NO !!
29     035376 032737 000100 001352 BIT #ECH,RMER1I ;IS ERROR CORRECTION HARD SET ?
30     035404 001053 BNE 80$ ;YES !!
31     035406 032737 010000 001370 BIT #FMT16,RMOFI ;IS THIS 16 BIT FORMAT ??
32     035414 001447 BEQ 80$ ;NO !!
33
34     ;CORRECT DATA USING ECC INFORMATION
35     035416 013700 001416 MOV RMBAD,R0 ;R0 = STARTING BUFFER ADDRESS
36     035422 013701 001402 MOV RMECI,R1 ;R1 = ECC POSITION
37     035426 052737 100000 035702 BIS #BIT15,150$ ;SET CORRECTION FLAG
38
39     ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST
40     10$: CMP #16.,R1 ;IS BIT POSITION > 1 WORD
41     BHS 20$ ;NO !!
42     SUB #16.,R1 ;SUBTRACT 1 WORDS WORTH
43     TST (R0)+ ;ADVANCE BUFFER ADDRESS 1 WORD
44     BR 10$
45
46     20$: MOV #1,R2 ;R2 = BIT POINTER
47     MOV R2,R3 ;R3 = BIT NUMBER
48
49     ;MOVE R2 TO STARTING BIT OF ERROR BURST
50     30$: CMP R3,R1 ;IS R3 SAME AS R1 ??
51     BEQ 35$ ;YES !!
52     ASL R2 ;SHIFT BIT POINTER
53     INC R3 ;INCREMENT BIT NUMBER
54     BR 30$
55
56     35$: MOV #11.,R3 ;R3 = LENGTH OF ERROR BURST
57
58     ;CORRECT THE ERROR BURST
59     40$: BIT R2,RMEC2I ;IS THIS BIT SET IN ECC PATTERN ??

```

```

54 035502 001405      BEQ      60$      ;NO - DO NOT CORRECT THIS BIT
55 035504 030210      BIT      R2,(R0)   ;IS THE BIT PRESENTLY SET ??
56 035506 001402      BEQ      50$      ;NO
57 035510 040210      BIC      R2,(R0)   ;RESET THE BIT
58 035512 000401      BR       60$
59 035514 050210      50$:  BIS      R2,(R0)   ;SET THE BIT
60 035516 006302      60$:  ASL      R2      ;SHIFT TO NEXT BIT
61 035520 001003      BNE      70$
62 035522 012702 000001  MOV      #1,R2     ;CONTINUE WITH FIRST BIT OF NEXT WORD
63 035526 005720      TST      (R0)+
64 035530 005303      70$:  DEC      R3     ;END OF BURST ??
65 035532 001361      BNE      40$      ;NO !
66
67      ;COMPARE WRITE BUFFER TO READ BUFFER
68 035534 017600 000010  80$:  MOV      @10(SP),R0 ;R0 = WRITE BUFFER
69 035540 062766 000002 000010  ADD      #2,10(SP)   ;MOVE SP TO READ ADDRESS
70 035546 017601 000010  MOV      @10(SP),R1 ;R1 = READ BUFFER
71 035552 062766 000002 000010  ADD      #2,10(SP)   ;MOVE SP TO RETURN ADDRESS
72 035560 013702 001340  MOV      RMWCI,R2   ;R2 = NUMBER OF WORDS TRANSFER
73 035564 163702 001414  SUB      RMWCO,R2
74 035570 022021      90$:  CMP      (R0)+,(R1)+ ;COMPARE DATA WORDS
75 035572 001003      BNE      100$      ;EXIT IF NOT EQUAL
76 035574 005302      DEC      R2        ;DECREMENT WORD COUNT
77 035576 001374      BNE      90$      ;CONTINUE IF NOT DONE
78 035600 000433      BR       110$     ;DONE COMPARE - NO ERROR
79
80      ;DATA COMPARE FAILED
81 035602 014037 001140  100$:  MOV      -(R0), $GDDAT ;STORE GOOD DATA FOR TYPEOUT
82 035606 014137 001142  MOV      -(R1), $BDDAT ;STORE BAD DATA FOR TYPEOUT
83 035612 010037 001134  MOV      R0,$GDADR   ;STORE ADDRESS OF GOOD DATA
84 035616 010137 001136  MOV      R1,$BDADR   ;STORE ADDRESS OF BAD DATA
85 035622 010237 001174  MOV      R2,$TMP0    ;STORE WORD COUNT OF ERROR
86 035626 062766 000004 000010  ADD      #4,10(SP)   ;MOVE SP TO USER'S ERROR CALL
87 035634 112776 000336 000010  MOVB    #336,@10(SP) ;WRITE ERROR NUMBER IN CALL
88
89      ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
90 035642 032737 100000 035702  BIT      #BIT15,150$ ;WAS ECC CORRECTION USED ??
91 035650 001403      BEQ      105$      ;NO !!
92 035652 112776 000163 000010  MOVB    #163,@10(SP) ;ECC CORRECTION FAILED
93 035660 162766 000002 000010  105$:  SUB      #2,10(SP)   ;MOVE SP TO RETURN IF ERROR
94 035666 000240      NOP
95 035670      110$:
96 035670 012603      MOV      (SP)+,R3   ;;POP STACK INTO R3
97 035672 012602      MOV      (SP)+,R2   ;;POP STACK INTO R2
98 035674 012601      MOV      (SP)+,R1   ;;POP STACK INTO R1
99 035676 012600      MOV      (SP)+,R0   ;;POP STACK INTO R0
100 035700 000207      RTS      PC        ;RETURN TO USER
101
102 035702 000000      150$:  .WORD      ;ECC CORRECTION FLAG

```

.SBTTL GET STATUS SUBROUTINE

:THIS SUBROUTINE SETS UP THE 'GET INDEX TABLE' AND THE 'GET
:BUFFER' FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
:AND THEN RETURNS TO THE USER.

:CALL: JSR PC,GETSTS
: ??? RETURN HERE

GETSTS:

10	035704			MOV	R0,-(SP)	::PUSH R0 ON STACK
	035704	010046		MOV	R1,-(SP)	::PUSH R1 ON STACK
11	035706	010146		MOV	R2,-(SP)	::PUSH R2 ON STACK
12	035710	010246		MOV	#GETINX,R0	:R0 = ADDRESS OF INDEX TABLE
13	035712	012700	001524	MOV	#RMEC2I+2,R1	:R1 = ADDRESS OF GET BUFFER
14	035716	012701	001406	MOV	#RMEC2,R2	:R2 = REGISTER INDE
15	035722	012702	000046	MOV	R2,(R0)+	:WRITE REGISTER INDEX IN TABLE
16	035726	110220		2\$:	CLR	-(R1)
17	035730	005041		3\$:	SUB	#2,R2
18	035732	162702	000002		BMI	4\$
19	035736	100405			CMR	#RMDB,R2
20	035740	022702	000022		BNE	2\$
21	035744	001370			CLR	-(R1)
22	035746	005041			BR	3\$
23	035750	000770		4\$:	MOVB	#200,(R0)+
24	035752	112720	000200		MOV	(SP)+,R2
25	035756	012602			MOV	(SP)+,R1
26	035760	012601			MOV	(SP)+,R0
27	035762	012600			NOP	
28	035764	000240			RTS	PC
29	035766	000207				

PROGRAM NAME

.SBTTL GET SUBROUTINE

:THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
 : "GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
 : LOCATION IN THE "GET REGISTER BUFFER". FOR EXAMPLE, AN
 : ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
 : READ "R4" AND STORE ITS CONTENTS AT THE LOCATION IN
 : THE BUFFER ASSIGNED TO THAT REGISTER. THE NUMBER OF
 : REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
 : TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
 : WHICH SHOULD FOLLOW THE LAST ENTRY.

:SUBROUTINE CALL:

- :(1) "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
VALUES AND TERMINATED WITH A CONTROL BYTE
- :(2) "GET INPUT BUFFER" IS AVAILABLE FOR USE. (NOTE THAT
UNUSED LOCATIONS, I.E., ENTRIES IN BUFFER CORRESPONDING
TO REGISTERS NOT READ, ARE NOT CHANGED.)
- :(3) JSR PC,GET
BR ??? RETURN HERE IF NO ERROR FOUND
NOP RETURN HERE IF ANY ERROR FOUND
ERROR SUB DEFINES ERROR NUMBER
???

:R0 REGISTER BASE ADDRESS
 :R1 REGISTER ADDRESS
 :R2 BUFFER BASE ADDRESS
 :R3 BUFFER ADDRESS
 :R4 POINTER TO REGISTER INDEX

```

GET:  NOP
      ADD    #4,(SP)      ;CLEAR ERROR NUMBER IN USER'S
      CLRB  @ (SP)      ;ERROR CALL
      SUB   #4,(SP)
      MOV   R0,-(SP)     ;:PUSH R0 ON STACK
      MOV   R1,-(SP)     ;:PUSH R1 ON STACK
      MOV   R2,-(SP)     ;:PUSH R2 ON STACK
      MOV   R3,-(SP)     ;:PUSH R3 ON STACK
      MOV   R4,-(SP)     ;:PUSH R4 ON STACK
      MOV   ERRVEC,-(SP) ;:PUSH ERRVEC ON STACK
      MOV   ERRVEC+2,-(SP) ;:PUSH ERRVEC+2 ON STACK
      MOV   $BASE,R0
      MOV   #GETBUF,R2
      MOV   #GETINX,R4
      MOV   #5$,ERRVEC   ;SETUP FOR TIMEOUT
      MOV   #PR6,ERRVEC+2
1$:   MOV   RMCS2(R0),$TMP0 ;GET 'NED' STATUS
      MOV   RMCS1(R0),$TMP1 ;GET 'DVA' STATUS
      BIT   #DVA,$TMP1    ;DEVICE AVAILABLE??
      BNE   3$           ;YES!!
      ADD   #4,16(SP)     ;WRITE ERROR NUMBER IN USER'S
      MOVB #112,@16(SP)  ;ERROR CALL
      BR    7$
3$:   TSTB  (R4)          ;DONE??
      BMI  9$           ;YES!!
      MOVB (R4),R1       ;R1 - REGISTER ADDRESS
      BIC  #*CIDXMSK,R1  ;CLEAR ANY SIGN EXTENSION
  
```

31	035770	000240		
32	035772	062716	000004	
33	035776	105076	000000	
34	036002	162716	000004	
35	036006	010046		
	036010	010146		
	036012	010246		
	036014	010346		
	036016	010446		
	036020	013746	000004	
	036024	013746	000006	
36	036030	013700	001276	
37	036034	012702	001336	
38	036040	012704	001524	
39	036044	012737	036152	000004
40	036052	012737	000300	000006
41	036060	016037	000010	001174
42	036066	016037	000000	001176
43	036074	032737	004000	001176
44	036102	001007		
45	036104	062766	000004	000016
46	036112	112776	000112	000016
47	036120	000423		
48	036122	105714		
49	036124	100433		
50	036126	111401		
51	036130	042701	177700	

```

52 036134 060001          ADD      R0,R1
53 036136 112403          MOVb    (R4)+,R3          ;R3 STORAGE ADDRESS FOR REGISTER
54 036140 042703 177700  BIC     #'CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
55 036144 060203          ADD     R2,R3
56 036146 011113          MOV     (R1),(R3)      ;READ REGISTER
57 036150 000764          BR     7$
58
59 036152 022626          5$:    CMP     (SP)+,(SP)+    ;RESTORE STACK
60 036154 062766 000004 000016  ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
61 036162 112776 000007 000016  MOVb    #7,@16(SP)     ;USER'S ERROR CALL
62 036170 162766 000002 000016  7$:    SUB     #2,16(SP)
63 036176 105714          8$:    TSTB   (R4)          ;DONE CLEARING??
64 036200 100405          BMI     9$            ;YES.!
65 036202 005003          CLR     R3           ;CLEAR REMAINING STORAGE
66 036204 112403          MOVb    (R4)+,R3     ;LOCATIONS
67 036206 060203          ADD     R2,R3
68 036210 005013          CLR     (R3)
69 036212 000771          BR     8$
70 036214          9$:
   036214 012637 000006          MOV     (SP)+,ERRVEC+2 ;:POP STACK INTO ERRVEC+2
   036220 012637 000004          MOV     (SP)+,ERRVEC  ;:POP STACK INTO ERRVEC
   036224 012604          MOV     (SP)+,R4      ;:POP STACK INTO R4
   036226 012603          MOV     (SP)+,R3      ;:POP STACK INTO R3
   036230 012602          MOV     (SP)+,R2      ;:POP STACK INTO R2
   036232 012601          MOV     (SP)+,R1      ;:POP STACK INTO R1
   036234 012600          MOV     (SP)+,R0      ;:POP STACK INTO R0
71 036236 000207          RTS     PC           ;RETURN

```

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

.SBTTL PUT SUBROUTINE

: THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
 : 'PUT INDEX TABLE' WITH THE CONTENTS OF THE CORRESPONDING
 : LOCATION IN THE 'PUT REGISTER BUFFER'. THE NUMBER OF
 : REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
 : BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
 : FOLLOW THE LAST ENTRY.

: SUBROUTINE CALL:

- : (1) 'PUT INDEX TABLE' HAS BEEN LOADED WITH INDEX VALUES
 : OF REGISTERS TO BE WRITTEN.
- : (2) 'PUT REGISTER BUFFER' CONTAINS CONTENTS OF EACH
 : REGISTER TO BE WRITTEN.
- : (3) JSR PC,PUT
 : BR ??? RETURN HERE IF NO ERROR FOUND
 : NOP RETURN HERE IF ANY ERROR FOUND
 : ERROR SUB DEFINES ERROR NUMBER
 : ???

:R0 - REGISTER BASE ADDRESS
 :R1 - REGISTER ADDRESS
 :R2 - BUFFER BASE ADDRESS
 :R3 - BUFFER ADDRESS
 :R4 - POINTER TO REGISTER INDEX

```

PUT:  NOP
      MOV R0,-(SP)      ;;PUSH R0 ON STACK
      MOV R1,-(SP)      ;;PUSH R1 ON STACK
      MOV R2,-(SP)      ;;PUSH R2 ON STACK
      MOV R3,-(SP)      ;;PUSH R3 ON STACK
      MOV R4,-(SP)      ;;PUSH R4 ON STACK
      MOV ERRVEC,-(SP)  ;;PUSH ERRVEC ON STACK
      MOV ERRVEC+2,-(SP) ;;PUSH ERRVEC+2 ON STACK
      MOV $BASE,R0
      MOV #PUTBUF,R2
      MOV #PUTINX,R4
      MOV #5$,ERRVEC    :SETUP FOR TIMEOUT
      MOV #PR6,ERRVEC+2
1$:   MOV RMCS2(R0),$TMP0 :GET 'NED' STATUS
      MOV RMCS1(R0),$TMP1 :GET 'DVA' STATUS
      BIT #DVA,$TMP1      :DEVICE AVAILABLE??
      BNE 3$             :YES!!
      ADD #4,16(SP)      :WRITE ERROR NUMBER IN
      MOVB #112,@16(SP) :USER'S ERROR CALL
      BR 7$
3$:   TSTB (R4)          :DONE??
      BMI 9$            :YES!!
      MOVB (R4),R1      :R1 = REGISTER ADDRESS
      BIC #^CIDXMSK,R1  :CLEAR ANY SIGN EXTENSION
      ADD R0,R1
      MOVB (R4),R3      :R3 = STORAGE ADDRESS
      BIC #^CIDXMSK,R3  :CLEAR ANY SIGN EXTENSION
      ADD R2,R3
      MOV (R3),(R1)     :WRITE REGISTER
4$:   TSTB (R4)+        :ADJUST REGISTER POINTER
  
```

```

28 036240 000240
29 036242 010046
   036244 010146
   036246 010246
   036250 010346
   036252 010446
   036254 013746 000004
   036260 013746 000006
30 036264 013700 001276
31 036270 012702 001412
32 036274 012704 001553
33 036300 012737 036410 000004
34 036306 012737 000300 000006
35 036314 016037 000010 001174
36 036322 016037 000000 001176
37 036330 032737 004000 001176
38 036336 001007
39 036340 062766 000004 000016
40 036346 112776 000112 000016
41 036354 000424
42 036356 105714
43 036360 100425
44 036362 111401
45 036364 042701 177700
46 036370 060001
47 036372 111403
48 036374 042703 177700
49 036400 060203
50 036402 011311
51 036404 105724
  
```

```

52 036406 000763          BR      3$
53
54 036410 022626          5$:    (MP      (SP)+,(SP)+      ;ADJUST STACK
55 036412 062766 000004 000016  ADD      #4,16(SP)      ;WRITE ERROR NUMBER IN
56 036420 112776 000007 000016  MOV      #7,@16(SP)    ;USER'S ERROR CALL
57 036426 162766 000002 000016  7$:    SUB      #2,16(SP)
58
59 036434          9$:
   036434 012637 000006  MOV      (SP)+,ERRVEC+2      ;:POP STACK INTO ERRVEC+2
   036440 012637 000004  MOV      (SP)+,ERRVEC      ;:POP STACK INTO ERRVEC
   036444 012604  MOV      (SP)+,R4          ;:POP STACK INTO R4
   036446 012603  MOV      (SP)+,R3          ;:POP STACK INTO R3
   036450 012602  MOV      (SP)+,R2          ;:POP STACK INTO R2
   036452 012601  MOV      (SP)+,R1          ;:POP STACK INTO R1
   036454 012600  MOV      (SP)+,R0          ;:POP STACK INTO R0
60 036456 000207  RTS      PC          ;RETURN
    
```

.SBTTL SIZE CLOCK SUBROUTINE

```

3 036460
4 036460 013746 000004
5 036464 013746 000006
6 036470 012737 036526 000004
7 036476 012737 000300 000006
8 036504 012737 177546 001520
9 036512 012737 000100 001522
10 036520 005777 142774
11 036524 000421
12 036526 022626
13 036530 012737 036560 000004
14 036536 012737 172540 001520
15 036544 012737 000104 001522
16 036552 005777 142742
17 036556 000404
18 036560 022626
19 036562 062766 000002 000004
20 036570
   036570 012637 000006
21 036574 012637 000004
22 036600 000207

SIZCLK:
MOV ERRVEC,-(SP)      ;;PUSH ERRVEC ON STACK
MOV ERRVEC+2,-(SP)   ;;PUSH ERRVEC+2 ON STACK
MOV #1$,ERRVEC       ;;SET UP FOR BUS TIMEOUT
MOV #PR6,ERRVEC+2
MOV #177546,CLKADR   ;;LOAD ADDRESSES FOR KW11-L
MOV #100,CLKVCT
TST @CLKADR          ;;TEST FOR KW11-L PRESENT
BR 3$                ;;YES - KW11-L IS PRESENT
1$: CMP (SP)+,(SP)+  ;;RESTORE SP
MOV #2$,ERRVEC       ;;SET UP FOR BUS TIMEOUT
MOV #172540,CLKADR   ;;LOAD ADDRESSES FOR KW11-P CLOCK
MOV #104,CLKVCT
TST @CLKADR          ;;TEST FOR KW11-P PRESENT
BR 3$                ;;YES - KW11-P IS PRESENT
2$: CMP (SP)+,(SP)+  ;;RESTORE SP
ADD #2,4(SP)         ;;MOVE RETURN TO ERROR
3$: MOV (SP)+,ERRVEC+2 ;;POP STACK INTO ERRVEC+2
MOV (SP)+,ERRVEC     ;;POP STACK INTO ERRVEC
RTS PC               ;;RETURN TO USER

```


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 036766
 42
 43
 44 036766 062716 -000004
 45 036772 105076 000000
 46 036776 162716 000004
 47
 48
 49 037002 013737 001346 001142
 50 037010 042737 177770 001142
 51 037016 013737 001234 001140
 52 037024 042737 177770 001140
 53 037032 123737 001140 001142
 54
 55 037040 001415
 56 037042 062716 000004
 57 037046 112776 000001 000000

```

.SBTTL ERROR CHECK SUBROUTINES
:*****
.SBTTL PRIMARY ERROR CHECK SUBROUTINE

:THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STATUS IS VALID AND
:THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED. THE
:FOLLOWING CHECKS ARE MADE:

:
:   .CORRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
:(BITS 0-2) EQUAL THE UNIT BEING TESTED;
:
:   .SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
:AND NED (BIT 12 OF RMCS2) IS RESET;
:
:   .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
:READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
:DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
:   .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
:I.E., MCPE = 0.
:   .NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
:I.E., PAR = 0, OR, PAR = DPE - 1

:
:THE SUBROUTINE ASSUMES THAT:

:
:   .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
:IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
:CORRESPONDING LOCATIONS OF THE 'GET' BUFFER.

:
:   .($UNIT) CONTAINS THE DRIVE NUMBER

:THE SUBROUTINE IS CALLED AS FOLLOWS:

:(1)   JSR    PC,PRIERR
:      BR     ???           RETURN HERE IF NO ERROR
:      NOP
:      ERROR  RETURN HERE TO REPORT AN ERROR
:      JSR    PC,@(SP)+    ERROR NUMBER DEFINED BY SUB
:      ???           GO BACK TO SUB FOR MORE ERROR CHECKS
:                   RETURN HERE IF NO MORE ERRORS

PRIERR:

:CLEAR USER'S ERROR CALL
ADD    #4,(SP)           ;MOVE (SP) TO ERROR CALL
CLRB   @(SP)           ;CLEAR ERROR NUMBER
SUB    #4,(SP)           ;MOVE (SP) TO NO ERROR RETURN

:REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
MOV    RMCS2I,$BDDAT    ;CORRECT UNIT SELECTED??
BIC    #^CUNTMSK,$BDDAT
MOV    $UNIT,$GDDAT     ;GOOD DATA FOR TYPEOUT
BIC    #^CUNTMSK,$GDDAT
CMPB   $GDDAT,$BDDAT    ;COMPARE EXPECTED AND RECEIVED
:DRIVE NUMBERS
BEQ    1$               ;YES.!
ADD    #4,(SP)
MOVB   #1,@(SP)        ;ERROR 1
  
```

```

58 037054 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
59 037060 004736                JSR      PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
60 037062 162716 000010          SUB      #10,(SP)        ;RESTORE (SP)
61 037066 000240                NOP
62 037070 000137 037610          JMP      10$             ;SKIP OTHER CHECKS
63 037074
64
65                                ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
66                                ;THE DEVICE IS NONEXISTANT
67 037074 032737 004000 001336    BIT      #DVA,RMCS1I     ;DEVICE AVAILABLE??
68 037102 001045                BNE      5$              ;YES!!
69 037104 013737 001336 001140    MOV      RMCS1I,$GDDAT   ;EXPECTED STATUS
70 037112 052737 004000 001140    BIS      #DVA,$GDDAT
71 037120 013737 001336 001142    MOV      RMCS1I,$BDDAT   ;RECEIVED STATUS
72 037126 062716 000004                ADD      #4,(SP)
73 037132 112776 000002 000000    MOVVB   #2,@(SP)        ;ERROR #2
74 037140 032737 010000 001346    BIT      #NED,RMCS2I     ;WAS NED SET??
75 037146 001414                BEQ      2$              ;NO!
76 037150 013737 001346 001140    MOV      RMCS2I,$GDDAT   ;EXPECTED STATUS
77 037156 013737 001346 001142    MOV      RMCS2I,$BDDAT   ;RECEIVED STATUS
78 037164 042737 010000 001140    BIC      #NED,$GDDAT
79 037172 112776 000003 000000    MOVVB   #3,@(SP)        ;YES - CHANGE ERROR NUMBER
80 037200 162716 000002          2$:    SUB      #2,(SP)        ;MOVE SP TO RETURN FOR ERROR
81 037204 004736                JSR      PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
82 037206 162716 000010          SUB      #10,(SP)        ;RESTORE (SP)
83 037212 000240                NOP
84 037214 000575                BR       10$             ;SKIP OTHER CHECKS
85 037216
86
87                                ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
88 037216 032737 000200 001336    BIT      #RDY,RMCS1I     ;CONTROLLER READY??
89 037224 001030                BNE      7$              ;YES!!
90 037226 013737 001336 001140    MOV      RMCS1I,$GDDAT   ;EXPECTED STATUS
91 037234 052737 000200 001140    BIS      #RDY,$GDDAT
92 037242 042737 160001 001140    BIC      #SC!TRE!MCPE!GO,$GDDAT
93 037250 013737 001336 001142    MOV      RMCS1I,$BDDAT   ;RECEIVED STATUS
94 037256 062716 000004                ADD      #4,(SP)
95 037262 112776 000004 000000    MOVVB   #4,@(SP)        ;ERROR #4
96 037270 162716 000002          SUB      #2,(SP)        ;MOVE SP TO RETURN FOR ERROR
97 037274 004736                JSR      PC,@(SP)+        ;REPORT CONTROLLER NOT READY
98 037276 162716 000010          SUB      #10,(SP)        ;RESTORE (SP)
99 037302 000240                NOP
100 037304 000541                BR       10$             ;SKIP OTHER CHECKS
101 037306
102
103                                ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
104 037306 032737 000001 001336    BIT      #GO,RMCS1I     ;GO RESET??
105 037314 001431                BEQ      8$              ;YES!!
106 037316 032737 000200 001350    BIT      #DRY,RMDSI     ;DRIVE READY??
107 037324 001025                BNE      8$              ;YES!
108 037326 013737 001336 001140    MOV      RMCS1I,$GDDAT   ;EXPECTED STATUS
109 037334 042737 160001 001140    BIC      #SC.TRE!MCPE!GO,$GDDAT
110 037342 013737 001336 001142    MOV      RMCS1I,$BDDAT   ;RECEIVED STATUS
111 037350 062716 000004                ADD      #4,(SP)
112 037354 112776 000005 000000    MOVVB   #5,@(SP)        ;ERROR #5
113 037362 162716 000002          SUB      #2,(SP)        ;MOVE SP TO RETURN FOR ERROR
114 037366 004736                JSR      PC,@(SP)+        ;REPORT DRIVE NOT READY

```

```

115 037370 162716 000010      SUB      #10,(SP)      ;RESTORE (SP)
116 037374 000240      NOP
117 037376 000504      BR       10$
118 037400
119
120
121      ;REPORT AN ERROR IF THE RM CONTROLLER DETECTED BAD
      ;PARITY ON THE MASSBUS CONTROL BUS
122 037400 032737 020000 001336      BIT      #MCPE,RMC5I1  ;PARITY ERROR ??
123 037406 001425      BEQ     9$           ;NO!!
124 037410 013737 001336 001140      MOV     RMC5I1,$GDDAT ;EXPECTED STATUS
125 037416 042737 160001 001140      BIC     #SC!TRÉ!MCPE.GO,$GDDAT
126 037424 013737 001336 001142      MOV     RMC5I1,$BDDAT ;RECEIVED STATUS
127 037432 062716 000004      ADD     #4,(SP)       ;MOVE STACK TO USER'S ERROR
128 037436 112776 000013 000000      MOVSB  #13,@(SP)     ;ERROR #13
129 037444 162716 000002      SUB     #2,(SP)       ;MOVE SP TO RETURN FOR ERROR
130 037450 004736      JSR     PC,@(SP)+    ;REPORT ERROR VIA USER
131 037452 162716 000010      SUB     #10,(SP)     ;RESTORE STACK
132 037456 000240      NOP
133 037460 000453      BR       10$
134 037462
135
136      ;REPORT AN ERROR IF DETECTED A CONTROL BUS PARITY ERROR
137 037462 032737 000010 001352      BIT      #PAR,RMER1I  ;WAS THERE A PARITY ERROR??
138 037470 001451      BEQ     11$         ;NO!!
139 037472 032737 000010 001400      BIT      #DPE,RMER2I  ;WAS IT THE CONTROL BUS??
140 037500 001045      BNE     11$         ;NOT SURE!!
141 037502 032737 000010 001426      BIT      #PAR,RMER1O  ;DID TEST SET PAR ??
142 037510 001413      BEQ     93$        ;NO!!
143 037512 010046      MOV     R0,-(SP)    ;:PUSH R0 ON STACK
144 037514 012700 001553      MOV     #PUTINX,R0   ;R0 POINTS TO INDEX TABLE
145 037520 122710 000014      CMPB   #RMER1,(R0)  ;SEARCH TABLE FOR RMER1
146 037524 001002      BNE     92$        ;
147 037526 012600      MOV     (SP)+,R0    ;:POP STACK INTO R0
148 037530 000431      BR      11$        ;PAR WAS SET BY TEST
149 037532 105720      TSTB   (R0)+       ;END OF TABLE??
150 037534 100371      BPL     91$        ;NO!!
151 037536 012600      MOV     (SP)+,R0    ;:POP STACK INTO R0
152 037540 013737 001352 001140      MOV     RMER1I,$GDDAT ;EXPECTED STATUS
153 037546 042737 000010 001140      BIC     #PAR,$GDDAT
154 037554 013737 001352 001142      MOV     RMER1I,$BDDAT ;RECEIVED STATUS
155 037562 062716 000004      ADD     #4,(SP)     ;MOVE SP TO USER'S ERROR CALL
156 037566 112776 000050 000000      MOVSB  #50,@(SP)    ;WRITE THE ERROR NUMBER
157 037574 162716 000002      SUB     #2,(SP)     ;MOVE SP TO RETURN FOR ERROR
158 037600 004736      JSR     PC,@(SP)+    ;REPORT THE ERROR
159 037602 162716 000010      SUB     #10,(SP)    ;MOVE SP TO NO ERROR RETURN
160 037606 000240      NOP
161 037610 062716 000010      10$:   ADD     #10,(SP)   ;RETURN TO ERROR
162 037614 000240      11$:   NOP
163 037616 000207      RTS     PC
  
```

.SBTTL SECONDARY ERROR CHECK SUBROUTINE

; THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
 ; SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER CONTENTS. THESE
 ; ERRORS ARE DEEMED SECONDARY IN THAT THEY ARE NOT NECESSARILY
 ; ASSOCIATED WITH THE OPERATION BEING PERFORMED.
 ; WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES THE ERROR
 ; NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
 ; TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL. AFTER THE TEST ROUTINE
 ; MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
 ; OTHER ERRORS. WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
 ; RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.

```

:CALL: JSR    PC,SECERR
:      BR     ???          RETURN HERE IF NO ERROR
:      NOP
:      ERROR          RETURN HERE TO REPORT AN ERROR
:      JSR    PC,@(SP)+    ERROR NUMBER DEFINED BY SUB
:      ???          GO BACK TO SUB FOR MORE ERROR CHECKS
:                      RETURN HERE IF NO MORE ERRORS
    
```

;NOTE: THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
 ;INPUT REGISTER BUFFER.

SECERR:

```

:*****
:STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
MOV    RMCS10,515$    ;STORE FUNCTION CODE
BIC    #^C<F0!F1.F2!F3.F4>,515$
ADD    #4,(SP)        ;MOVE (SP) TO ERROR CALL
CLRB   @ (SP)         ;CLEAR ERROR NUMBER
SUB    #4,(SP)        ;MOVE (SP) TO NO ERROR RETURN
    
```

;*****
 ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS

```

:REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY 0
BIT    #DRY,RMDSI    ;DRIVE READY??
BNE    5$           ;YES!!
MOV    RMDSI,$BDDAT ;BAD DATA FOR TYPEOUT
BIC    #^CDRY,$BDDAT
MOV    #DRY,$GDDAT  ;GOOD DATA FOR TYPEOUT
ADD    #4,(SP)
MOVB   #10,@(SP)   ;ERROR NUMBER
SUB    #2,(SP)     ;MOVE SP TO RETURN FOR ERROR
JSR    PC,@(SP)+  ;REPORT NOT READY
SUB    #10,(SP)   ;RESTORE (SP) TO ERROR N
NOP
    
```

```

:REPORT ERROR IF GO BIT IS NOT RESET
5$: BIT    #GO,RMCS11 ;GO BIT RESET??
BEQ    10$          ;YES!!
MOV    RMCS11,$BDDAT ;BAD DATA FOR TYPEOUT
BIC    #^CGO,$BDDAT
CLR    $GDDAT      ;GOOD DATA FOR TYPEOUT
ADD    #4,(SP)
MOVB   #11,@(SP)  ;ERROR NUMBER
    
```

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24 037620
25
26
27
28 037620 013737 001412 043454
29 037626 042737 177701 043454
30 037634 062716 000004
31 037640 105076 000000
32 037644 162716 000004
33
34
35
36
37
38 037650 032737 000200 001350
39 037656 001024
40 037660 013737 001350 001142
41 037666 042737 177577 001142
42 037674 012737 000200 001140
43 037702 062716 000004
44 037706 112776 000010 000000
45 037714 162716 000002
46 037720 004736
47 037722 162716 000010
48 037726 000240
49
50
51 037730 032737 000001 001336
52 037736 001423
53 037740 013737 001336 001142
54 037746 042737 177776 001142
55 037754 005037 001140
56 037760 062716 000004
57 037764 112776 000011 000000
    
```

```

58 037772 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
59 037776 004736 JSR PC,@(SP)+ ;REPORT DEVICE NOT AVAILABLE
60 040000 162716 000010 SUB #10,(SP) ;RESTORE (SP)
61 040004 000240 NOP
62
63 ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
64 040006 013737 001336 001142 10$: MOV RMCS1I,$BDDAT ;IS FUNCTION CODE CORRECT??
65 040014 042737 177701 001142 BIC #^C76,$BDDAT
66 040022 013737 043454 001140 MOV 515$,$GDDAT ;EXPECTED FUNCTION CODE
67 040030 023737 001142 001140 CMP $BDDAT,$GDDAT
68 040036 001413 BEQ 15$ ;YES..
69 040040 062716 000004 ADD #4,(SP)
70 040044 112776 000012 000000 MOVB #12,@(SP) ;ERROR NUMBER
71 040052 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
72 040056 004736 JSR PC,@(SP)+ ;REPORT WRONG FUNCTION CODE
73 040060 162716 000010 SUB #10,(SP) ;RESTORE (SP)
74 040064 000240 NOP
75 040066
76
77 ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
78 ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
;OTHER ERRORS ARE SET
79 040066 005037 001140 CLR $GDDAT ;EXPECT 'ERR' = 0
80 040072 005737 001352 TST RMER1I ;IS RMER1 = 0??
81 040076 001003 BNE 20$ ;NO!!
82 040100 005737 001400 TST RMER2I ;IS RMERZ = 0??
83 040104 001403 BEQ 25$ ;YES!!
84 040106 052737 040000 001140 20$: BIS #ERR,$GDDAT ;'ERR' SHOULD BE SET
85 040114 013737 001350 001142 25$: MOV RMDSI,$BDDAT
86 040122 042737 137777 001142 BIC #^CERR,$BDDAT
87 040130 023737 001140 001142 CMP $GDDAT,$BDDAT ;IS 'ERR' OK??
88 040136 001412 BEQ 30$ ;YES!!
89 040140 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
90 040144 112776 000047 000000 MOVB #47,@(SP) ;WRITE ERROR NUMBER
91 040152 162716 000002 SUB #2,(SP) ;MOVE SP TO ERROR RETURN
92 040156 004736 JSR PC,@(SP)+ ;REPORT INVALID COMP ERROR
93 040160 162716 000010 SUB #10,(SP)
94
95 ;REPORT AN ERROR IF 'TRE' IS SET AND NONE OF THE BITS WHICH SET
96 ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
97 ;SET TRE IS SET
98 040164 005037 001140 30$: CLR $GDDAT ;EXPECT 'TRE' = 0
99 040170 013746 001346 MOV RMCS2I,-(SP) ;WAS DLT, WCE, UPE, NED, NEM
100 040174 042726 000377 BIC #377,(SP)+ ;PGE, MXF OR MDPE SET
101 040200 001010 BNE 35$ ;YES!!
102 040202 032737 040000 001350 BIT #ERR,RMDSI ;WAS EXCEPTION RECEIVED??
103 040210 001407 BEQ 40$ ;NO!!
104 040212 022737 000030 043454 CMP #SEARCH,515$ ;WAS DATA TRANSFERRED??
105 040220 103003 BHIS 40$ ;NO!!
106 040222 052737 040000 001140 35$: BIS #TRE,$GDDAT ;'TRE' SHOULD BE SET
107 040230 013737 001336 001142 40$: MOV RMCS1I,$BDDAT ;BAD DATA FOR TYPEOUT
108 040236 042737 137777 001142 BIC #^CTRE,$BDDAT
109 040244 023737 001140 001142 CMP $GDDAT,$BDDAT ;IS 'TRE' OK??
110 040252 001413 BEQ 45$ ;YES!!
111 040254 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
112 040260 112776 000014 000000 MOVB #14,@(SP) ;WRITE ERROR NUMBER
113 040266 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
114 040272 004736 JSR PC,@(SP)+ ;REPORT TRE ERROR

```

```

115 040274 162716 000010          SUB      #10,(SP)          ;RESTORE (SP)
116 040300 000240          NOP
117 040302          45$:
118
119          ;*****
120          ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
121          ;      .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
122          ;      .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
123          ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
124          ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
125          ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
126
127          ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
128 040302 010046          MOV      RO,-(SP)          ;;PUSH RO ON STACK
129 040304 013700 043454          MOV      515$,RO          ;GET FUNCTION CODE
130 040310 016037 064566 043446          MOV      FNCDTB(RO),500$ ;STORE ENTRY
131 040316 012600          MOV      (SP)+,RO          ;;POP STACK INTO RO
132
133          ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
134          ;ATA IS NOT SET AND SHOULD BE SET.
135 040320 013737 043446 001140          MOV      500$,$GDDAT      ;GET EXPECTED ATA STATUS
136 040326 032737 040000 001350          BIT      #ERR,RMDSI      ;IS COMPOSITE ERROR SET ??
137 040334 001403          BEQ      50$           ;NO !
138 040336 052737 100000 001140          BIS      #ATA,$GDDAT      ;EXPECT AN ATTENTION
139 040344 042737 077777 001140 50$:          BIC      #^CATA,$GDDAT     ;STRIP DONT CARES
140 040352 013737 001350 001142          MOV      RMDSI,$BDDAT     ;GET RECEIVED ATA
141 040360 042737 077777 001142          BIC      #^CATA,$BDDAT     ;STRIP DONT CARES
142 040366 023737 001140 001142          CMP      $GDDAT,$BDDAT    ;IS ATA OK ??
143 040374 001413          BEQ      55$           ;YES !!
144 040376 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
145 040402 112776 000006 000000          MOVB     #6,@(SP)         ;LOAD ERROR # IN CALL
146 040410 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
147 040414 004736          JSR      PC,@(SP)+        ;REPORT ERROR
148 040416 162716 000010          SUB      #10,(SP)         ;RESTORE SP
149 040422 000240          NOP
150 040424          55$:
151
152          ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
153          ;WITH FUNCTION CODE TABLE
154 040424 013737 043446 001140          MOV      500$,$GDDAT      ;GET EXPECTED ILF
155 040432 042737 177776 001140          BIC      #^CILF,$GDDAT     ;CLEAR ALL OTHER BITS
156 040440 013737 001352 001142          MOV      RMER1I,$BDDAT     ;GET RECEIVED ILF
157 040446 042737 177776 001142          BIC      #^CILF,$BDDAT     ;CLEAR ALL OTHER BITS
158 040454 023737 001140 001142          CMP      $GDDAT,$BDDAT    ;IS ILF OK ??
159 040462 001412          BEQ      60$           ;YES !!
160 040464 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
161 040470 112776 000254 000000          MOVB     #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
162 040476 162716 000002          SUB      #2,(SP)          ;MOVE SP TO ERROR RETURN
163 040502 004736          JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
164 040504 162716 000010          SUB      #10,(SP)         ;MOVE SP TO NO ERROR
165 040510 005037 001140 60$:          CLR      $GDDAT          ;CLEAR EXPECTED STATUS
166
167          ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
168 040514 013746 043446          MOV      500$,-(SP)       ;GET WCE STATUS ENABLE
169 040520 052716 137777          BIS      #^(WCE,(SP)      ;SET ALL OTHER BITS
170 040524 013737 001346 001142          MOV      RMCS2I,$BDDAT     ;RECEIVED STATUS
171 040532 042637 001142          BIC      (SP)+,$BDDAT      ;CLEAR WCE IF ENABLED
    
```

```

SECONDARY ERROR CHECK SUBROUTINE

172 040536 001412          BEQ      90$          :BRANCH IF WCE OK
173 040540 062716 C00004    ADD      #4,(SP)      :MOVE SP TO USER'S ERROR CALL
174 040544 112776 000026 000000    MOVB    #26,@(SP)    :WRITE ERROR NUMBER
175 040552 162716 000002          SUB      #2,(SP)      :MOVE SP TO ERROR RETURN
176 040556 004736          JSR     PC,@(SP)+    :REPORT ERROR
177 040560 162716 000010          SUB      #10,(SP)     :RESTORE ERROR
178 040564          90$:
179
180          :REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SFT
181 040564 013746 043446    MOV     500$,-(SP)   :GET OPI STATUS ENABLE
182 040570 052716 157777    BIS    #^COPI,(SP)  :SET ALL OTHER BITS
183 040574 013737 001352 001142    MOV     RMER1I,$BDDAT :GET RECEIVED STATUS
184 040602 042637 001142          BIC    (SP)+,$BDDAT  :CLEAR OPI IF ENABLED
185 040606 001412          BEQ     100$         :BRANCH IF OPI OK
186 040610 062716 000004    ADD     #4,(SP)      :MOVE SP TO USER'S ERROR CALL
187 040614 112776 000164 000000    MOVB    #164,@(SP)   :WRITE ERROR NUMBER IN CALL
188 040622 162716 000002          SUB     #2,(SP)      :MOVE SP TO ERROR RETURN
189 040626 004736          JSR     PC,@(SP)+    :REPORT ERROR
190 040630 162716 000010          SUB     #10,(SP)     :RESTORE SP
191 040634          100$:
192
193          :REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
194          :SET AND VV IS NOT RESET
195 040634 013746 043446    MOV     500$,-(SP)   :GET IVC STATUS ENABLE
196 040640 032737 000100 001350    BIT     #VV,RMDSI    :IS VV SET
197 040646 001402          BEQ     105$         :NO !!
198 040650 042716 010000          BIC    #IVC,(SP)    :YES - IVC SHOULD BE 0
199 040654 052716 167777 105$:    BIS    #^CIVC,(SP)  :SET ALL OTHER BITS
200 040660 013737 001400 001142    MOV     RMER2I,$BDDAT :GET RECEIVED STATUS
201 040666 042637 001142          BIC    (SP)+,$BDDAT  :CLEAR IVC IF ENABLED
202 040672 001412          BEQ     110$         :BRANCH IF IVC OK
203 040674 062716 000004    ADD     #4,(SP)      :MOVE SP TO USERS ERROR CALL
204 040700 112776 000165 000000    MOVB    #165,@(SP)   :WRITE ERROR NUMBER IN CALL
205 040706 162716 000002          SUB     #2,(SP)      :MOVE SP TO ERROR RETURN
206 040712 004736          JSR     PC,@(SP)+    :REPORT ERROR
207 040714 162716 000010          SUB     #10,(SP)     :RESTORE SP TO NO ERROR
208 040720          110$:
209
210          :BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
211          :ALL WRITE ERRORS, I.E.,
212          :   RMER1 - WLE, WCF
213          :   RMER2 - DPE
214          :   RMCS2 - UPE.
215          :EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
216          :WRITE ERROR ENABLE BIT IS RESET.
217
218          :REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
219          :THE DRIVE IS NOT WRITE PROTECTED
220 040720 012746 177777    MOV     #-1,-(SP)    :ASSUME WRITE ERRORS ENABLED
221 040724 032737 004000 043446    BIT     #WLE,500$    :ARE WRITE ERRORS ENABLED ??
222 040732 001404          BEQ     115$         :NO !!
223 040734 032737 004000 001350    BIT     #WRL,RMDSI   :IS THE DRIVE WRITE PROTECTED ??
224 040742 001002          BNE     120$         :YES !!
225 040744 042716 004000 115$:    BIC    #WLE,(SP)    :RESET WLE ENABLE
226 040750 013737 001352 001142 120$:    MOV     RMER1I,$BDDAT :GET RECEIVED STATUS
227 040756 042637 001142          BIC    (SP)+,$BDDAT  :CLEAR WLE IF ENABLED
228 040762 001412          BEQ     125$         :BRANCH IF WLE OK

```


229	040764	062716	000004			ADD	#4,(SP)	:MOVE SP TO USERS ERROR CALL
230	040770	112776	000023	000000		MOV	#2,@(SP)	:WRITE ERROR NUMBER IN CALL
231	040776	162716	000002			SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
232	041002	004736				JSR	PC,@(SP)+	:REPORT ERROR AND RETURN
233	041004	162716	000010			SUB	#10,(SP)	:RESTORE SP TO NO ERROR
234	041010				125\$:			
235								
236						:REPORT ERROR IF WCF IS SET AND		WRITE ERRORS ARE NOT ENABLED
237	041010	012746	177777			MOV	#-1,-(SP)	:ASSUME WRITE ERRORS ENABLED
238	041014	032737	004000	043446		BIT	#WLE,500\$:ARE WRITE ERRORS ENABLED ??
239	041022	001002				BNE	130\$:YES !!
240	041024	042716	000040			BIC	#WCF,(SP)	:DISABLE WCF ERROR
241	041030	013737	001352	001142	130\$:	MOV	RMER1I,\$BDDAT	:GET RECEIVED STATUS
242	041036	042637	001142			BIC	(SP)+,\$BDDAT	:RESET WCF IF ENABLED
243	041042	001412				BEQ	135\$:BRANCH IF WCF OK
244	041044	062716	000004			ADD	#4,(SP)	:MOVE SP TO USERS ERROR CALL
245	041050	112776	000025	000000		MOV	#25,@(SP)	:WRITE ERROR NUMBER IN CALL
246	041056	162716	000002			SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
247	041062	004736				JSR	PC,@(SP)+	:REPORT ERROR
248	041064	162716	000010			SUB	#10,(SP)	:RESTORE SP TO NO ERROR
249	041070				135\$:			
250								
251						:REPORT ERROR IF DPE IS SET AND		WRITE ERRORS ARE NOT ENABLED
252	041070	012746	177777			MOV	#-1,-(SP)	:ASSUME WRITE ERRORS ARE ENABLED
253	041074	032737	004000	043446		BIT	#WLE,500\$:ARE WRITE ERRORS ENABLED ??
254	041102	001002				BNE	140\$:YES !
255	041104	042716	000010			BIC	#DPE,(SP)	:RESET DPE ENABLE
256	041110	013737	001400	001142	140\$:	MOV	RMER2I,\$BDDAT	:GET RECEIVED STATUS
257	041116	042637	001142			BIC	(SP)+,\$BDDAT	:RESET DPE IF ENABLED
258	041122	001412				BEQ	145\$:BRANCH IF DPE OK
259	041124	062716	000004			ADD	#4,(SP)	:MOVE SP TO USERS ERROR CALL
260	041130	112776	000040	000000		MOV	#40,@(SP)	:WRITE ERROR NUMBER IN CALL
261	041136	162716	000002			SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
262	041142	004736				JSR	PC,@(SP)+	:REPORT ERROR
263	041144	162716	000010			SUB	#10,(SP)	:RESTORE SP TO NO ERROR
264	041150				145\$:			
265								
266						:REPORT AN ERROR IF UPE IS SET AND		WRITE ERRORS ARE NOT ENABLED
267	041150	012746	177777			MOV	#-1,-(SP)	:ASSUME WRITE ERRORS ARE ENABLED
268	041154	032737	004000	043446		BIT	#WLE,500\$:ARE WRITE ERRORS ENABLED ??
269	041162	001002				BNE	150\$:YES !
270	041164	042716	020000			BIC	#UPE,(SP)	:DISABLE UPE ERROR
271	041170	013737	001346	001142	150\$:	MOV	RMCS2I,\$BDDAT	:GET RECEIVED STATUS
272	041176	042637	001142			BIC	(SP)+,\$BDDAT	:RESET UPE IF ENABLED
273	041202	001412				BEQ	155\$:BRANCH IF UPE OK
274	041204	062716	000004			ADD	#4,(SP)	:MOVE SP TO USERS ERROR CALL
275	041210	112776	000024	000000		MOV	#24,@(SP)	:WRITE ERROR NUMBER IN CALL
276	041216	162716	000002			SUB	#2,(SP)	:MOVE SP TO ERROR RETURN
277	041222	004736				JSR	PC,@(SP)+	:REPORT ERROR AND RETURN
278	041224	162716	000010			SUB	#10,(SP)	:MOVE SP TO NO ERROR
279	041230				155\$:			
280								
281						:REPORT AN ERROR IF IAE IS SET AND		IS NOT ENABLED
282	041230	013746	043446			MOV	500\$,-(SP)	:GET IAE ENABLE
283	041234	052716	175777			BIS	#^CIAE,(SP)	:SET ALL OTHER BITS
284	041240	013737	001352	001142		MOV	RMER1I,\$BDDAT	:GET RECEIVED STATUS
285	041246	042637	001142			BIC	(SP)+,\$BDDAT	:CLEAR IAE IF ENABLED

286	041252	001412			BEQ	160\$:BRANCH IF IAE IS OK
287	041254	062716	000004		ADD	#4,(SP)		:MOVE SP TO USERS ERROR CALL
288	041260	112776	000166	000000	MOVB	#166,@(SP)		:WRITE ERROR NUMBER
289	041266	162716	000002		SUB	#2,(SP)		:MOVE SP TO ERROR RETURN
290	041272	004736			JSR	PC,@(SP)+		:REPORT ERROR AND RETURN
291	041274	162716	000010		SUB	#10,(SP)		:MOVE SP TO NO ERROR
292	041300					160\$:		
293								
294								:BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
295								: ALL READ/WRITE ERRORS, I.E.,
296								: RMCS1 - TRE
297								: RMCS2 - DLT,NEM,MXF
298								: RMDS - LBT
299								: RMER1 - AOE
300								:NOTE:
301								: LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
302								:CYLINDER REGISTER IS WRITTEN
303								:NOTE:
304								: AOE CANNOT BE SET IF LBT IS NOT ALSO SET
305								:NOTE:
306								: TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITIONS ABOVE
307								
308								
309								:REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
310	041300	012746	177777		MOV	#-1,-(SP)		:ASSUME ERRORS ARE ENABLED
311	041304	032737	001000	043446	BIT	#AOE,500\$:ARE ERRORS ENABLED ??
312	041312	001002			BNE	165\$:YES !!
313	041314	042716	100000		BIC	#DLT,(SP)		:RESET DLT ENABLE
314	041320	013737	001346	001142	165\$:	MOV	RMCS2I,\$BDDAT	:GET RECEIVED STATUS
315	041326	042637	001142		BIC	(SP)+,\$BDDAT		:CLEAR DLT IF ENABLED
316	041332	001412			BEQ	170\$:BRANCH IF DLT IS OK
317	041334	062716	000004		ADD	#4,(SP)		:MOVE SP TO USERS ERROR CALL
318	041340	12776	000032	000000	MOVB	#32,@(SP)		:WRITE ERROR NUMBER IN CALL
319	041346	162716	000002		SUB	#2,(SP)		:MOVE SP TO ERROR RETURN
320	041352	004736			JSR	PC,@(SP)+		:REPORT ERROR AND RETURN
321	041354	162716	000010		SUB	#10,(SP)		:MOVE SP TO NO ERROR
322	041360					170\$:		
323								
324								:REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
325	041360	012746	177777		MOV	#-1,-(SP)		:ASSUME ERRORS ARE ENABLED
326	041364	032737	001000	043446	BIT	#AOE,500\$:ARE ERRORS ENABLED ??
327	041372	001002			BNE	175\$:YES !!
328	041374	042716	004000		BIC	#NEM,(SP)		:DISABLE NEM
329	041400	013737	001346	001142	175\$:	MOV	RMCS2I,\$BDDAT	:GET RECEIVED STATUS
330	041406	042637	001142		BIC	(SP)+,\$BDDAT		:CLEAR NEM IF ENABLED
331	041412	001412			BEQ	180\$:BRANCH IF NEM IS OK
332	041414	062716	000004		ADD	#4,(SP)		:MOVE SP TO USERS ERROR CALL
333	041420	112776	000167	000000	MOVB	#167,@(SP)		:WRITE ERROR NUMBER IN CALL
334	041426	162716	000002		SUB	#2,(SP)		:MOVE SP TO ERROR RETURN
335	041432	004736			JSR	PC,@(SP)+		:REPORT ERROR AND RETURN
336	041434	162716	000010		SUB	#10,(SP)		:MOVE SP TO NO ERROR
337	041440					180\$:		
338								
339								:REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
340	041440	012746	177777		MOV	#-1,-(SP)		:ASSUME ERRORS ARE ENABLED
341	041444	032737	001000	043446	BIT	#AOE,500\$:ARE DATA ERRORS ENABLED ??
342	041452	001002			BNE	185\$:YES

```

343 041454 042716 001000          BIC    #MXF,(SP)      :DISABLE MXF ERROR
344 041460 013737 001346 001142 18$:  MOV    RMCS2I,$BDDAT  :GET RECEIVED STATUS
345 041466 042637 001142          BIC    (SP)+,$BDDAT  :CLEAR MXF IF ENABLED
346 041472 001412          BEQ    190$          :BRANCH IF MXF IS OK
347 041474 062716 000004          ADD    #4,(SP)       :MOVE SP TO USERS ERROR CALL
348 041500 112776 000033 000000  MOVB   #33,@(SP)     :WRITE ERROR NUMBER IN CALL
349 041506 162716 000002          SUB    #2,(SP)       :MOVE SP TO ERROR RETURN
350 041512 004736          JSR    PC,@(SP)+     :REPORT ERROR AND RETURN
351 041514 162716 000010          SUB    #10,(SP)      :MOVE SP TO NO ERROR
352 041520          190$:
353
354          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
355 041520 012746 177777          MOV    #-1,-(SP)    :ASSUME DATA ERRORS ARE ENABLED
356 041524 032737 001000 043446  BIT    #AOE,500$    :ARE DATA ERRORS ENABLED ??
357 041532 001404          BEQ    191$          :NO !!
358 041534 032737 002000 001350  BIT    #LBT,RMDS!   :IS LBT ALSO SET ??
359 041542 001002          BNE    195$          :YES !!
360 041544 042716 001000 191$:  BIC    #AOE,(SP)    :DISABLE AOE
361 041550 013737 001352 001142 195$:  MOV    RMER1I,$BDDAT :GET RECEIVED STATUS
362 041556 042637 001142          BIC    (SP)+,$BDDAT :CLEAR AOE IF ENABLED
363 041562 001412          BEQ    200$          :BRANCH IF AOE IS OK
364 041564 062716 000004          ADD    #4,(SP)       :MOVE SP TO USERS ERROR CALL
365 041570 112776 000020 000000  MOVB   #20,@(SP)    :WRITE ERROR NUMBER
366 041576 162716 000002          SUB    #2,(SP)       :MOVE SP TO ERROR RETURN
367 041602 004736          JSR    PC,@(SP)+     :REPORT ERROR AND RETURN
368 041604 162716 000010          SUB    #10,(SP)      :MOVE SP TO NO ERROR
369 041610          200$:
370
371          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
372          ;HEADER ERRORS, I.E.,
373          ;   RMER1 - HCRC,HCE,FER
374          ;   RMER2 - BSE
375
376          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
377 041610 032737 002000 001370  BIT    #HCI,RMOFI   :IS HCI SET ??
378 041616 001403          BEQ    201$          :NO !!
379 041620 042737 000200 043446  BIC    #HCE,500$    :YES - DISABLE ALL HEADER ERRORS
380 041626          201$:
381
382          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
383 041626 012746 177777          MOV    #-1,-(SP)    :ASSUME ERRORS ENABLED
384 041632 032737 000200 043446  BIT    #HCE,500$    :ARE HEADER ERRORS ENABLED ??
385 041640 001002          BNE    205$          :YES !!
386 041642 042716 000400          BIC    #HCRC,(SP)   :DISABLE HCRC
387 041646 013737 001352 001142 205$:  MOV    RMER1I,$BDDAT :GET RECEIVED STATUS
388 041654 042637 001142          BIC    (SP)+,$BDDAT :RESET HCRC IF ENABLED
389 041660 001412          BEQ    210$          :BRANCH IF HCRC IS OK
390 041662 062716 000004          ADD    #4,(SP)       :MOVE SP TO USERS ERROR CALL
391 041666 112776 000035 000000  MOVB   #35,@(SP)    :WRITE ERROR NUMBER IN CALL
392 041674 162716 000002          SUB    #2,(SP)       :MOVE SP TO ERROR RETURN
393 041700 004736          JSR    PC,@(SP)+     :REPORT ERROR AND RETURN
394 041702 162716 000010          SUB    #10,(SP)      :MOVE SP TO NO ERROR
395 041706          210$:
396
397          ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
398 041706 012746 177777          MOV    #-1,-(SP)    :ASSUME ERRORS ENABLED
399 041712 032737 000200 043446  BIT    #HCE,500$    :ARE ERRORS ENABLED ??
    
```

```

400 041720 001002          BNE      215$      :YES !!
401 041722 042716 000200    BIC      #HCE,(SP)  :DISABLE HCE
402 041726 013737 001352    001142  215$:  MOV      RMER1I,$BDDAT :GET RECEIVED STATUS
403 041734 042637 001142    BIC      (SP)+,$BDDAT :CLEAR HCE IF ENABLED
404 041740 001412          BEQ      220$      :BRANCH IF HCE IS OK
405 041742 062716 000004    ADD      #4,(SP)    :MOVE SP TO USERS ERROR CALL
406 041746 112776 000036    000000  MOVVB   #36,@(SP)   :WRITE ERROR NUMBER IN CALL
407 041754 162716 000002    SUB      #2,(SP)    :MOVE SP TO ERROR RETURN
408 041760 004736          JSR      PC,@(SP)+  :REPORT ERROR AND RETURN
409 041762 162716 000010    SUB      #10,(SP)   :MOVE SP TO NO ERROR
410 041766          220$:
411
412          ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
413 041766 012746 177777    MOV      #-1,-(SP)  :ASSUME FER IS ENABLED
414 041772 032737 000200    043446  BIT      #HCE,500$  :ARE HEADER ERRORS ENABLED ??
415 042000 001002          BNE      225$      :YES !!
416 042002 042716 000020    BIC      #FER,(SP)  :DISABLE FER
417 042006 013737 001352    001142  225$:  MOV      RMER1I,$BDDAT :GET RECEIVED STATUS
418 042014 042637 001142    BIC      (SP)+,$BDDAT :RESET FER IF ENABLED
419 042020 001412          BEQ      230$      :BRANCH IF FER OK
420 042022 062716 000004    ADD      #4,(SP)    :MOVE SP TO USERS ERROR CALL
421 042026 112776 000037    000000  MOVVB   #37,@(SP)   :WRITE ERROR NUMBER IN CALL
422 042034 162716 000002    SUB      #2,(SP)    :MOVE SP TO ERROR RETURN
423 042040 004736          JSR      PC,@(SP)+  :REPORT ERROR AND RETURN
424 042042 162716 000010    SUB      #10,(SP)   :MOVE SP TO NO ERROR
425 042046          230$:
426
427          ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
428 042046 012746 177777    MOV      #-1,-(SP)  :ASSUME ERRORS ENABLED
429 042052 032737 000200    043446  BIT      #HCE,500$  :ARE THEY ENABLED ??
430 042060 001002          BNE      235$      :YES !!
431 042062 042716 100000    BIC      #BSE,(SP)  :DISABLE BSE
432 042066 013737 001400    001142  235$:  MOV      RMER2I,$BDDAT :GET RECEIVED STATUS
433 042074 042637 001142    BIC      (SP)+,$BDDAT :CLEAR BSE IF ENABLED
434 042100 001412          BEQ      240$      :BRANCH IF BSE OK
435 042102 062716 000004    ADD      #4,(SP)    :MOVE SP TO USERS ERROR CALL
436 042106 112776 000354    000000  MOVVB   #354,@(SP)  :WRITE ERROR NUMBER
437 042114 162716 000002    SUB      #2,(SP)    :MOVE SP TO ERROR RETURN
438 042120 004736          JSR      PC,@(SP)+  :REPORT ERROR AND RETURN
439 042122 162716 000010    SUB      #10,(SP)   :MOVE SP TO NO ERROR
440 042126          240$:
441
442          ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
443          ;FIELD ERRORS, I.E.,
444          ;      RMCS2 - MDPE
445          ;      RMER1 - DCK,ECH
446          ;NOTE:
447          ;      ECH CANNOT SET UNLESS IT IS ENABLED AND FCI IS RESET AND
448          ;      DCK IS SET.
449
450          ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
451 042126 012746 177777    MOV      #-1,-(SP)  :ASSUME ENABLED
452 042132 032737 000100    043446  BIT      #ECH,500$  :ARE DATA FIELD ERRORS ENABLED ??
453 042140 001002          BNE      245$      :YES !!
454 042142 042716 000400    BIC      #MDPE,(SP) :DISBALE MDPE
455 042146 013737 001346    001142  245$:  MOV      RMCS2I,$BDDAT :GET RECEIVED STATUS
456 042154 042637 001142    BIC      (SP)+,$BDDAT :CLEAR MDPE IF ENABLED

```

457	042160	001412			BEQ	250\$:BRANCH IF MDPE OK
458	042162	062716	000004		ADD	#4,(SP)		:MOVE SP TO USERS ERROR CALL
459	042166	112776	000027	000000	MOVB	#27,@(SP)		:WRITE ERROR NUMBER IN CALL
460	042174	162716	000002		SUB	#2,(SP)		:MOVE SP TO ERROR RETURN
461	042200	004736			JSR	PC,@(SP)+		:REPORT ERROR AND RETURN
462	042202	162716	000010		SUB	#10,(SP)		:MOVE SP TO NO ERROR
463	042206					250\$:		
464								
465								:REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
466	042206	012746	177777		MOV	#-1,-(SP)		:ASSUME ENABLED
467	042212	032737	000100	043446	BIT	#ECH,500\$:ARE THEY ENABLED ??
468	042220	001002			BNE	255\$:YES !!
469	042222	042716	100000		BIC	#DCK,(SP)		:DISABLE DCK
470	042226	013737	001352	001142	255\$:	MOV	RMER11,\$BDDAT	:GET RECEIVED STATUS
471	042234	042637	001142		BIC	(SP)+,\$BDDAT		:CLEAR DCK IF ENABLED
472	042240	001412			BEQ	260\$:BRANCH IF DCK IS OK
473	042242	062716	000004		ADD	#4,(SP)		:MOVE SP TO USERS ERROR CALL
474	042246	112776	000030	000000	MOVB	#30,@(SP)		:WRITE ERROR NUMBER IN CALL
475	042254	162716	000002		SUB	#2,(SP)		:MOVE SP TO ERROR RETURN
476	042260	004736			JSR	PC,@(SP)+		:REPORT ERROR AND RETURN
477	042262	162716	000010		SUB	#10,(SP)		:MOVE SP TO NO ERROR
478	042266					260\$:		
479								
480								:REPORT ERROR IF ECH IS SET AND,
481								:DATA FIELD ERRORS ARE NOT ENABLED, OR
482								:ECI IS SET, OR
483								:DCK IS NOT SET.
484	042266	012746	177777		MOV	#-1,-(SP)		:ASSUME ENABLED
485	042272	032737	000100	043446	BIT	#ECH,500\$:ARE ERRORS ENABLED ??
486	042300	001410			BEQ	265\$:NO !!
487	042302	032737	004000	001370	BIT	#ECI,RMOFI		:IS ECI SET ??
488	042310	001004			BNE	265\$:YES !!
489	042312	032737	100000	001352	BIT	#DCK,RMER11		:IS DCK ALSO SET ??
490	042320	001002			BNE	270\$:YES !!
491	042322	042716	000100		265\$:	BIC	#ECH,(SP)	:DISABLE ECH
492	042326	013737	001352	001142	270\$:	MOV	RMER11,\$BDDAT	:GET RECEIVED STATUS
493	042334	042637	001142		BIC	(SP)+,\$BDDAT		:CLEAR ECH IF ENABLED
494	042340	001412			BEQ	275\$:BRANCH IF ECH IS OK
495	042342	062716	000004		ADD	#4,(SP)		:MOVE SP TO USERS ERROR CALL
496	042346	112776	000031	000000	MOVB	#31,@(SP)		:WRITE ERROR NUMBER IN CALL
497	042354	162716	000002		SUB	#2,(SP)		:MOVE SP TO ERROR RETURN
498	042360	004736			JSR	PC,@(SP)+		:REPORT ERROR AND RETURN
499	042362	162716	000010		SUB	#10,(SP)		:MOVE SP TO NO ERROR
500	042366					275\$:		

```

1
2
3
4
5 042366 022737 000030 043454      CMP      #SEARCH,515$      ;WAS DATA TRANSFERRED ?
6 042374 103402      BLO      280$             ;BR IF YES
7 042376 000137 043420      JMP      355$             ;NO - EXIT
8
9
10 042402 013737 001340 001142      ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
11 042410 001421      280$:  MOV      RMWCI,$BDDAT      ;WORD COUNT ZERO??
12 042412 032737 040000 001336      BEQ      285$             ;YES
13 042420 001015      BIT      #TRE,RMCS11      ;TRANSFER ERROR DETECTED??
14 042422 062716 000004      BNE      285$             ;YES.!!
15 042426 112776 000015 000000      ADD      #4,(SP)          ;ERROR NUMBER
16 042434 005037 001140      MOV      #15,@(SP)        ;GOOD DATA FOR TYPEOUT
17 042440 162716 000002      CLR      $GDDAT           ;MOVE SP TO RETURN FOR ERROR
18 042444 004736      JSR      PC,@(SP)+        ;REPORT WORD COUNT NOT ZERO
19 042446 162716 000010      SUB      #10,(SP)         ;RESTORE (SP)
20 042452 000240      NOP
21
22
23 042454 013737 001340 001140      ;REPORT ERROR IF RMBA IS NOT CORRECT
24 042462 163737 001414 001140      285$:  MOV      RMWCI,$GDDAT      ;GET WORD COUNT AT END OF TRANSFER AND
25 042470 006337 001140      SUB      RMWCO,$GDDAT      ;SUBTRACT STARTING WORD COUNT.
26 042474 063737 001416 001140      ASL      $GDDAT           ;* 2
27
28 042502 032737 000010 001346      ADD      RMBAO,$GDDAT      ;ADD STARTING BUS ADDRESS
29 042510 001403      BIT      #BAI,RMCS2I      ;WAS BUS ADDRESS INHIBIT (BAI) SET ??
30 042512 013737 001416 001140      BEQ      290$             ;NO !!
31
32 042520 023737 001140 001342      MOV      RMBAO,$GDDAT      ;ADDRESS SHOULD NOT HAVE CHANGED
33 042526 001416      290$:  CMP      $GDDAT,RMBAI      ;BUS ADDRESS OK??
34 042530 013737 001342 001142      BEQ      295$             ;YES!!
35 042536 062716 000004      MOV      RMBAI,$BDDAT      ;BAD DATA FOR TYPEOUT
36 042542 112776 000016 000000      ADD      #4,(SP)          ;ERROR NUMBER
37 042550 162716 000002      MOV      #16,@(SP)        ;MOVE SP TO RETURN FOR ERROR
38 042554 004736      JSR      PC,@(SP)+        ;REPORT UNEXPECTED ADDRESS
39 042556 162716 000010      SUB      #10,(SP)         ;RESTORE (SP)
40 042562 000240      NOP
41
42
43 042564 005046      ;COMPUTE NUMBER OF SECTORS TRANSFERRED FROM WORD COUNT
44 042566 013746 001340      295$:  CLR      -(SP)           ;NUMBER OF SECTORS TRANSFERRED
45 042572 163716 001414      MOV      RMWCI,-(SP)       ;GET WORD COUNT AT END OF TRANSFER AND
46
47 042576 012746 000400      SUB      RMWCO,(SP)       ;SUBTRACT STARTING WORD COUNT.
48 042602 032737 000002 001412      MOV      #256,-(SP)       ;ASSUME 256. WORDS PER SECTOR
49 042610 001402      BIT      #BIT1,RMCS10     ;HEADER & DATA COMMAND ??
50 042612 062716 000002      BEQ      300$             ;NO !!
51
52 042616 005266 000004      ADD      #2,(SP)          ;CHANGE TO 258. WORDS PER SECTOR
53 042622 161666 000002      300$:  INC      4(SP)           ;INCREMENT SECTOR COUNT
54 042626 003373      SUB      (SP),2(SP)       ;SUBTRACT ONE SECTOR'S WORTH
55 042630 022626      BGT      300$             ;CONTINUE IF NOT DONE
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
    ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM
    
```

```

58r
59 042632 013737 001446 043446      :NUMBER OF SECTORS
60 042640 013737 001420 043450      MOV      RMDCO,500$      ;STORE ORIGINAL CYLINDER
61 042646 013737 001420 043452      MOV      RMDAO,505$      ;STORE ORIGINAL TRACK
62 042654 013737 001334 043456      MOV      RMDAO,510$      ;STORE ORIGINAL SECTOR
63 042662 000337 043456      MOV      LSTRK,520$      ;STORE LAST TRACK
64 042666 005237 043456      SWAB     520$            ;GET TRACK ADDRESS TO LO BYTE AND
65                                     INC      520$            ;INCREMENT TO GET TOTL # OF TRACKS.
66 042672 042737 000377 043450      BIC      #^C<TADMSK>,505$ ;SAVE TRACK ADDRESS BITS AND
67 042700 000337 043450      SWAB     505$            ;SWAP TRACK ADDRESS TO LOW BYTE.
68 042704 042737 177400 043452      BIC      #^C<SADMSK>,510$ ;SAVE SECTOR ADDRESS BITS
69 042712 062637 043452      ADD      (SP)+,510$
70
71 042716 023727 043452 000040 310$:  CMP      510$,#32.      ;SECTOR OVEFLOWED??
72 042724 103406      BLO      315$          ;NO!!
73 042726 005237 043450      INC      505$          ;INCREMENT TRACK
74 042732 162737 000040 043452      SUB      #32.,510$     ;ADJUST SECTOR
75 042740 000766      BR       310$          ;TRY AGAIN
76
77 042742 023737 043450 043456 315$:  CMP      505$,520$     ;TRACK OVERFLOWED??
78 042750 103407      BLO      320$          ;NO!!
79 042752 005237 043446      INC      500$          ;INCREMENT CYLINDER
80 042756 163737 043456 043450      SUB      520$,505$     ;ADJUST TRACK
81 042764 000766      BR       315$          ;TRY AGAIN
82 042766 000240      NOP
83
84                                     ;REPORT ERROR IF 'LBT' IS NOT CORRECT
85 042770      320$:
86 042770 005037 001140      CLR      $GDDAT        ;SET GOOD DATA FOR LBT 0
87 042774 023727 043446 001466      CMP      500$,#822.    ;SHOULD LBT BE SET??
88 043002 101407      BLOS     325$          ;NO!!
89 043004 032737 002000 001352      BIT      #IAE,RMER1I   ;WAS IAE SET ??
90 043012 001003      BNE      325$          ;YES - LBT SHOULD NOT BE SET
91 043014 012737 002000 001140      MOV      #LBT,$GDDAT   ;SET GOOD DATA FOR LBT 1
92 043022 013737 001350 001142 325$:  MOV      RMDSI,$BDDAT  ;BAD DATA FOR TYPEOUT
93 043030 042737 175777 001142      BIC      #^CLBT,$BDDAT
94 043036 023737 001140 001142      CMP      $GDDAT,$BDDAT ;IS LBT CORRECT??
95 043044 001413      BEQ     330$          ;YES!!
96 043046 062716 000004      ADD      #4,(SP)
97 043052 112776 000017 000000      MOVSB   #17,@(SP)     ;ERROR NUMBER
98 043060 162716 000002      SUB     #2,(SP)       ;MOVE SP TO RETURN FOR ERROR
99 043064 004736      JSR     PC,@(SP)+     ;REPORT LBT IS WRONG
100 043066 162716 000010      SUB     #10,(SP)      ;RESTORE (SP)
101 043072 000240      NOP
102
103                                     ;REPORT ERROR IF 'AOE' IS INCORRECT
104 043074 005037 001140      330$:  CLR      $GDDAT        ;SET FOR AOE - 0
105 043100 032737 002000 001352      BIT      #IAE,RMER1I   ;WAS 'IAE' DETECTED??
106 043106 001031      BNE     340$          ;YES-'AOE' SHOULD BE ZERO
107 043110 023727 043446 001466      CMP      500$,#822.    ;SHOULD AOE BE SET??
108 043116 101425      BLOS     340$          ;NO!!
109 043120 005737 043450      TST     505$          ;MAYBE
110 043124 001012      BNE     335$          ;YES
111 043126 005737 043452      TST     510$
112 043132 001007      BNE     335$          ;YES !!
113 043134 032737 000010 043454      BIT      #F2,515$     ;WAS THIS READ OR WRITE CHECK ??
114 043142 001413      BEQ     340$          ;NO !!
    
```

```

115 043144 005737 001340          TST      RMWCI          ;WAS ALL DATA TRANSFERRED ??
116 043150 001410          BEQ      340$          ;YES !!
117 043152 012737 001000 001140 335$: MOV      #AOE,$GDDAT    ;SET FOR AOE = 1
118 043160 005037 043450          CLR      505$          ;CLEAR EXPECTED TRACK
119 043164 012737 000001 043452          MOV      #1,510$       ;EXPECT SECTOR = 1
120 043172 013737 001352 001142 340$: MOV      RMER1I,$BDDAT ;BAD DATA FOR TYPEOUT
121 043200 042737 176777 001142          BIC      #^CAOE,$BDDAT
122 043206 023737 001140 001142          CMP      $GDDAT,$BDDAT ;IS AOE CORRECT??
123 043214 001413          BEQ      345$          ;YES!!
124 043216 062716 000004          ADD      #4,(SP)
125 043222 112776 000020 000000          MOVSB   #20,@(SP)      ;ERROR NUMBER
126 043230 162716 000002          SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
127 043234 004736          JSR      PC,@(SP)+    ;REPORT AOE IS WRONG
128 043236 162716 000010          SUB      #10,(SP)     ;RESTORE (SP)
129 043242 000240          NOP
130
131
132 043244 032737 002000 001352 345$: ;REPORT ERROR IF RMDA IS NOT CORRECT
133 043252 001062          BIT      #IAE,RMER1I  ;WAS THERE AN IAE ERROR ??
134 043254 013737 043450 001140          BNE      355$          ;YES - DONT CHECK RMDA,RMDC
135 043262 000337 001140          MOV      505$,$GDDAT ;SETUP EXPECTED DISK ADDRESS
136 043266 113737 043452 001140          SWAB    $GDDAT
137 043274 013737 001344 001142          MOVSB   510$,$GDDAT
138 043302 023737 001140 001142          MOV      RMDAI,$BDDAT ;SETUP RECEIVED DISK ADDRESS
139 043310 001413          CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED & RECEIVED
140 043312 062716 000004          BEQ      350$          ;BRANCH IF EQUAL
141 043316 112776 000021 000000          ADD      #4,(SP)
142 043324 162716 000002          MOVSB   #21,@(SP)     ;ERROR NUMBER
143 043330 004736          SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
144 043332 162716 000010          JSR      PC,@(SP)+    ;REPORT BAD DISK ADDRESS
145 043336 000240          SUB      #10,(SP)     ;RESTORE (SP)
146
147
148 043340 013737 043446 001140 350$: ;REPORT ERROR IF RMDC IS INCORRECT
149 043346 042737 176000 001140          MOV      500$,$GDDAT ;SETUP EXPECTED CYLINDER
150 043354 013737 001372 001142          BIC      #^C1777,$GDDAT
151 043362 023737 001140 001142          MOV      RMDCI,$BDDAT ;SETUP RECEIVED CYLINDER
152 043370 001413          CMP      $GDDAT,$BDDAT ;COMPARE CYLINDERS
153 043372 062716 000004          BEQ      355$          ;BRANCH IF EQUAL
154 043376 112776 000022 000000          ADD      #4,(SP)
155 043404 162716 000002          MOVSB   #22,@(SP)     ;ERROR NUMBER
156 043410 004736          SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
157 043412 162716 000010          JSR      PC,@(SP)+    ;REPORT BAD CYLINDER
158 043416 000240          SUB      #10,(SP)     ;RESTORE (SP)
159
160 043420 062716 000004          355$: ADD      #4,(SP)      ;MOVE (SP) TO ERROR CALL
161 043424 105776 000000          TSTB    @(SP)         ;WAS ERROR FOUND??
162 043430 001403          BEQ      360$
163 043432 062716 000004          ADD      #4,(SP)      ;MOVE (SP) TO ERROR RETURN
164 043436 000402          BR      365$
165 043440 162716 000004          360$: SUB      #4,(SP)      ;MOVE (SP) TO NO ERROR RETURN
166 043444 000207          365$: RTS      PC
167
168 043446 000000          500$: .WORD    0        ;CYLINDER
169 043450 000000          505$: .WORD    0        ;TRACK
170 043452 000000          510$: .WORD    0        ;SECTOR
171 043454 000000          515$: .WORD    0        ;FUNCTION CODE
    
```


*72 043456 000000

5208: .WORD 0

:TOTAL # OF TRACKS LAST TRACK *

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

.SBTTL COMPOSITE ERROR CHECK SUBROUTINE

: THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
 : RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
 : MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
 : THE MASKS ARE APPLIED.

: CALL:
 : (1) JSR PC, CMPERRSTS
 : .WORD MASK FOR ERROR REGISTER 1
 : .WORD MASK FOR ERROR REGISTER 2
 : BR ??? RETURN HERE IF NO ERROR
 : NOP RETURN HERE TO REPORT AN ERROR
 : ERROR ERROR NUMBER DEFINED BY SUB
 : JSR PC, @(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
 : ??? RETURN HERE IF NO MORE ERRORS

: NOTE: BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
 : BE ZERO

CMPERRSTS:

: MASK AND STORE THE CONTENTS OF RMER1 AND RMER2
 MOV RMER1, \$TMP1 ; STORE RMER1 AT TEMP STORAGE
 BIC @(SP), \$TMP1 ; MASK RMER1
 ADD #2, (SP) ; MOVE SP TO NEXT MASK
 MOV RMER2, \$TMP2 ; STORE RMER2 AT TEMP STORAGE
 BIC @(SP), \$TMP2 ; MASK RMER2

: CLEAR USER'S ERROR CALL
 ADD #6, (SP) ; MOVE SP TO USER'S ERROR CALL
 CLRB @(SP) ; CLEAR ERROR NUMBER
 SUB #4, (SP) ; LEAVE SP AT NO ERROR RETURN

: SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., \$TMP1
 TST \$TMP1 ; ANY ERRORS TO REPORT??
 BEQ 5\$; NO !!
 MOV \$TMP1, \$BDDAT ; RECEIVED STATUS FOR TYPEOUT
 CLR \$GDDAT ; EXPECTED STATUS FOR TYPEOUT
 ADD #4, (SP) ; MOVE SP TO USER'S ERROR CALL
 MOV #66, @(SP) ; CORRECTABLE DATA CHECK ERROR #
 SUB #2, (SP) ; MOVE SP TO RETURN FOR ERROR
 JSR PC, @(SP)+ ; REPORT ERROR VIA USER
 SUB #10, (SP) ; MOVE SP BACK TO BRANCH
 NOP

5\$:

: SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 (\$TMP2)
 TST \$TMP2 ; ANY ERRORS IN RMER2?
 BEQ 10\$; NO !!
 MOV \$TMP2, \$BDDAT ; RECEIVED STATUS FOR TYPEOUT
 CLR \$GDDAT ; EXPECTED STATUS FOR TYPEOUT
 ADD #4, (SP) ; MOVE SP TO USER'S ERROR CALL
 MOV #67, @(SP) ; WRITE ERROR NUMBER IN USER'S CALL
 SUB #2, (SP) ; MOVE SP TO RETURN FOR ERROR

043460
 043460 013737 001352 001176
 043466 047637 000000 001176
 043474 062716 000002
 043500 013737 001400 001200
 043506 047637 000000 001200
 043514 062716 000006
 043520 105076 000000
 043524 162716 000004
 043530 005737 001176
 043534 001420
 043536 013737 001176 001142
 043544 005037 001140
 043550 062716 000004
 043554 112776 000066 000000
 043562 162716 000002
 043566 004736
 043570 162716 000010
 043574 000240
 043576
 043576 005737 001200
 043602 001420
 043604 013737 001200 001142
 043612 005037 001140
 043616 062716 000004
 043622 112776 000067 000000
 043630 162716 000002

```

58 043634 004736      JSR      PC,@(SP)+      ;REPORT ERROR VIA USER
59 043636 162716 000010  SUB      #10,(SP)        ;MOVE SP TO NO ERROR RETURN
60 043642 000240      NOP
61 043644
62
63
64 043644 062716 000004  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
65 043650 105776 000000  ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
66 043654 001403      TSTB     @ (SP)         ;WAS THERE AN ERROR CALLED??
67 043656 062716 000004  BEQ      20$            ;NO. !
68 043662 000402      ADD      #4,(SP)        ;YES - MOVE SP TO ERROR RETURN
69 043664 162716 000004  BR       30$
70 043670 000207      SUB      #4,(SP)        ;MOVE SP TO NO ERROR RETURN
                          RTS      PC          ;RETURN TO USER
20$:
30$:

```

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

.SBTTL DEVICE SELECT SUBROUTINE

; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
 ; TEST QUEUE.

```

:CALL:
:(1) JSR PC,DEVSEL
:(2) BR ?? RETURN IF NO ERROR
:(3) NOP RETURN IF ERROR
:(4) ERROR ERROR DEFINED BY SUBROUTINE
    
```

DEVSEL:

```

;CLEAR USER'S ERROR CALL
ADD #4,(SP) ;MOVE SP TO USER'S ERROR
CLRB @10(SP) ;CLEAR LOW ORDER BYTE OF CALL
SUB #4,(SP) ;MOVE SP BACK
    
```

```

;SAVE USER'S INFORMATION AND SETUP REGISTERS
MOV ERRVEC,-(SP) ;;PUSH ERRVEC ON STACK
MOV ERRVEC+2,-(SP) ;;PUSH ERRVEC+2 ON STACK
MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV #20$,ERRVEC ;SETUP FOR BUS TIMEOUT
MOV #PR6,ERRVEC+2
MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
MOV TSTQUE,R1 ;R1 POINTS TO DEVICE NUMBER
    
```

```

;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
MOVB (R1),RMCS2(R0) ;WRITE UNIT SELECT BITS
MOV RMCS1(R0),$TMP1 ;GET 'DVA' STATUS
MOV RMCS2(R0),$TMP0 ;GET 'NED' STATUS
BIT #NED,$TMP0 ;IS DEVICE NONEXISTENT ?
BEQ 10$ ;NO!!
ADD #4,10(SP) ;MOVE SP TO USERS ERROR CALL
MOVB #111,@10(SP) ;WRITE ERROR NUMBER
BR 30$
    
```

```

10$: BIT #DVA,$TMP1 ;IS DEVICE AVAILABLE ?
BNE 35$ ;YES!!
ADD #4,10(SP) ;MOVE SP TO USERS ERROR CALL
MOVB #112,@10(SP) ;WRITE ERROR NUMBER
BR 30$
    
```

```

;HANDLE BUS TIMEOUT
20$: CMP (SP)+,(SP)+ ;ADJUST SP
ADD #4,10(SP) ;MOVE SP TO USERS ERROR CALL
MOVB #113,@10(SP) ;WRITE BUS TIMEOUT ERROR NUMBER
30$: SUB #2,10(SP) ;ADJUST RETURN TO 'NOP' PRECEDING
;THE ERROR CALL
    
```

```

;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
35$: MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
MOV (SP)+,ERRVEC+2 ;;POP STACK INTO ERRVEC+2
    
```

```

043672
043672 062716 000004
043676 105076 000000
043702 162716 000004
043706 013746 000004
043712 013746 000006
043716 010046
043720 010146
043722 012737 044042 000004
043730 012737 000300 000006
043736 013700 001276
043742 013701 001466
043746 111160 000010
043752 016037 000000 001176
043760 016037 000010 001174
043766 032737 010000 001174
043774 001407
043776 062766 000004 000010
044004 112776 000111 000010
044012 000422
044014 032737 004000 001176
044022 001021
044024 062766 000004 000010
044032 112776 000112 000010
044040 000407
044042 022626
044044 062766 000004 000010
044052 112776 000113 000010
044060 162766 000002 000010
044066
044066 012601
044070 012600
044072 012637 000006
    
```

044076 012637 000004
S2 044102 000207

MOV (SP)+,ERRVEC ::POP STACK INTO ERRVEC
RTS PC :EXIT

.SBTTL SEEK STATUS CHECK SUBROUTINE

:THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
:STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.

:THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
:AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE 'ERROR' TRAP
:OF THE CALLING ROUTINE. SEEK STATUS IS CHECKED AS FOLLOWS:

```
:CALL:
:(1) JSR    PC,SEKSTS
      BR     ???           RETURN HERE IF NO ERROR
      NOP           RETURN HERE TO REPORT AN ERROR
      ERROR        ERROR NUMBER DEFINED BY SUB
      JSR    PC,@(SP)+    GO BACK TO SUB FOR MORE ERROR CHECKS
      ???           RETURN HERE IF NO MORE ERRORS
```

SEKSTS:

:CLEAR USERS' ERROR CALL

```
NOP
ADD    #4,(SP)           ;MOVE (SP) TO ERROR CALL
CLRB   @(SP)            ;CLEAR ERROR NUMBER
SUB    #4,(SP)           ;MOVE (SP) TO NO ERROR RETURN
CLR    300$             ;CLEAR ERROR FLAGS
```

:TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
:LOCAL REGISTERS, I.E., "PAR" - 1 AND "DPE" - 0

```
BIT    #PAR,RMER1I      ;WAS PARITY ERROR DETECTED??
BEQ    1$               ;NO!!
BJT    #DPE,RMER2I      ;WAS IT DUE TO CONTROL BUS??
BNE    1$               ;NOT SURE.!
```

:REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL

```
CLR    $GDDAT           ;EXPECTED STATUS
MOV    RMER1I,$BDDAT    ;RECEIVED STATUS
ADD    #4,(SP)           ;MOVE STACK TO USER'S ERROR
MOVB   #50,@(SP)        ;ERROR #50
SUB    #2,(SP)           ;MOVE SP TO RETURN FOR ERROR
JSR    PC,@(SP)+
SUB    #10,(SP)         ;RESTORE STACK
BR     3$               ;IAE SHOULD BE ZERO
```

:DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK, SECTOR AND CYLINDER
:ALSO, SET "SKI" IF CYLINDER ADDRESS IS TOO LARGE.

```
1$: MOV    #IAE,$GDDAT   ;SETUP FOR IAE - 1
     BIS    #SKI,300$    ;SETUP FOR SKI = 1
     CMP    RMDCO,#822.  ;GREATER THAN LAST CYLINDER ?
     BHI   3$           ;YES - CYLINDER IS INVALID
     BIC   #SKI,300$    ;CLEAR SKI ERROR FLAG

     CMPB   RMDAO+1,LSTRK+1 ;GREATER THAN LAST TRACK ?
     BHI   3$           ;YES - TRACK IS INVALID

     CMPB   RMDAO,#29.   ;SECTOR > 29. ?
     BLOS  2$           ;BR IF NO
```

```

2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19 044104
20
21
22 044104 000240
23 044106 062716 000004
24 044112 105076 000000
25 044116 162716 000004
26 044122 005037 045342
27
28
29
30 044126 032737 000010 001352
31 044134 001424
32 044136 032737 000010 001400
33 044144 001020
34
35
36 044146 005037 001140
37 044152 013737 001352 001142
38 044160 062716 000004
39 044164 112776 000050 000000
40 044172 162716 000002
41 044176 004736
42 044200 162716 000010
43 044204 000437
44
45
46
47 044206 012737 002000 001140
48 044214 052737 040000 045342
49 044222 023727 001446 001466
50 044230 101025
51 044232 042737 040000 045342
52
53 044240 123737 001421 001335
54 044246 101016
55
56 044250 123727 001420 000035
57 044256 101410
```

```

58 044260 032737 010000 001444 BIT #FMT16,RMOFO ;18 BIT FORMAT ?
59 044266 001406 BEQ 3$ ;YES - SECTOR IS INVALID FOR 18 BIT MODE
60 044270 123727 001420 000037 CMPB RMDAO,#31. ;SECTOR > 31. ?
61 044276 101002 BHI 3$ ;YES - SECTOR IS INVALID
62
63 044300 005037 001140 2$: CLR $GDDAT ;"IAE" SHOULD = 0
64
65 ;COMPARE EXPECTED AND RECIEVED "IAE" STATUS
66 044304 013737 001352 001142 3$: MOV RMER1I,$BDDAT ;IS IAE OK??
67 044312 042737 175777 001142 BIC #^CIAE,$BDDAT ;SAVE IAE BIT FOR COMPARE
68 044320 023737 001140 001142 CMP $GDDAT,$BDDAT ;CORRECT "IAE" STATUS ?
69 044326 001004 BNE 35$ ;BR IF NO
70 044330 042737 040000 045342 BIC #SKI,300$ ;CLEAR SKI FLAG
71 044336 000413 BR 5$ ;GO CHECK NEXT ERROR
72 044340
73 35$:
74 044340 062716 000004 ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
75 044344 112776 000051 000000 ADD #4,(SP)
76 044352 162716 000002 MOVB #51,@(SP) ;ERROR 51
77 044356 004736 JSR PC,@(SP)+ ;MOVE SP TO RETURN FOR ERROR
78 044360 162716 000010 SUB #10,(SP) ;REPORT INCORRECT IAE
79 044364 000240 NOP ;RESTORE (SP)
80 044366
81 5$:
82 ;REPORT ANY IVC ERROR AS
83 ; IVC ERROR WITH VOLUME VALID ZERO
84 ; ERRONEOUS IVC ERROR, VOLUME VALID IS SET
85 044366 032737 010000 001400 BIT #IVC,RMER2I ;IVC ERROR??
86 044374 001427 BEQ 52$ ;NO!!
87 044376 005037 001140 CLR $GDDAT ;EXPECTED STATUS
88 044402 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
89 044410 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
90 044414 112776 000060 000000 MOVB #60,@(SP) ;ERROR 60 IF VV = 0
91 044422 032737 000100 001350 BIT #VV,RMDSI
92 044430 001403 BEQ 51$
93 044432 112776 000061 000000 MOVB #61,@(SP) ;ERROR 61 IF VV = 1
94 044440 162716 000002 51$: SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
95 044444 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
96 044446 162716 000010 SUB #10,(SP) ;RESTORE SP
97 044452 000240 NOP
98
99 044454 013737 001400 001142 52$: MOV RMER2I,$BDDAT ;RECEIVED STATUS
100 044462 042737 137777 001142 BIC #^CSKI,$BDDAT ;CLEAR DONT CARES
101 044470 013737 045342 001140 MOV 300$,$GDDAT ;GET EXPECTED SKI STATUS
102 044476 042737 137777 001140 BIC #^CSKI,$GDDAT ;CLEAR DONT CARES
103 044504 001417 BEQ 53$ ;BRANCH IF 0 EXPECTED
104
105 ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
106 044506 032737 040000 001142 BIT #SKI,$BDDAT ;WAS SKI DETECTED ??
107 044514 001032 BNE 54$ ;YES !!
108 044516 062716 000004 ADD #4,(SP) ;MOVE SP TO USERS ERROR CALL
109 044522 112776 000267 000000 MOVB #267,@(SP) ;WRITE ERROR NUMBER
110 044530 162716 000002 SUB #2,(SP) ;MOVE SP TO ERROR RETURN
111 044534 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
112 044536 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
113 044542 000443 BR 6$ ;GO TO NEXT ERROR CHECK
114 044544 53$:

```

```

115
116 ;REPORT ERROR IF SKI IS SET
117 044544 032737 040000 001142 BIT #SKI,$BDDAT ;IS SKI SET ??
118 044552 001413 BEQ 54$ ;NO - SKI IS OK
119 044554 062716 000004 ADD #4,(SP) ;MOVE (SP) TO ERROR
120 044560 112776 000054 000000 MOVB #54,@(SP) ;LOAD ERROR NUMBER
121 044566 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
122 044572 004736 JSR PC,@(SP)+ ;REPORT SEEK ERROR
123 044574 162716 000010 SUB #10,(SP) ;RESTORE (SP)
124 044600 000240 NOP
125
126 ;REPORT ANY DEVICE CHECK
127 044602 032737 000200 001400 54$: BIT #DVC,$RMER2I ;WAS THERE DVC DURING SEEK??
128 044610 001420 BEQ 6$ ;NO.
129 044612 005037 001140 CLR $GDDAT ;EXPECTED STATUS
130 044616 013737 001400 001142 MOV $RMER2I,$BDDAT ;RECEIVED STATUS
131 044624 062716 000004 ADD #4,(SP)
132 044630 112776 000055 000000 MOVB #55,@(SP) ;ERROR #55
133 044636 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
134 044642 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
135 044644 162716 000010 SUB #10,(SP) ;RESTORE SP
136 044650 000240 NOP
137
138 ;REPORT ANY 'OPI' ERROR AS OPI WITH MOL 0, OR OPI
139 ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
140 044652 032737 020000 001352 6$: BIT #OPI,$RMER1I ;'OPI' ERROR??
141 044660 001427 BEQ 8$ ;NO.
142 044662 005037 001140 CLR $GDDAT ;EXPECTED STATUS
143 044666 013737 001352 001142 MOV $RMER1I,$BDDAT ;RECEIVED STATUS
144 044674 062716 000004 ADD #4,(SP) ;MOVE (SP) TO ERROR
145 044700 112776 000052 000000 MOVB #52,@(SP) ;LOAD ERROR NUMBER
146 044706 032737 010000 001350 BIT #MOL,$RMDSI ;WAS MEDIUM ON LINE??
147 044714 001403 BEQ 7$ ;NO!!
148 044716 112776 000053 000000 MOVB #53,@(SP) ;YES - CHANGE ERROR NUMBER
149 044724 162716 000002 7$: SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
150 044730 004736 JSR PC,@(SP)+ ;REPORT 'OPI' ERROR
151 044732 162716 000010 SUB #10,(SP) ;RESTORE (SP)
152 044736 000240 NOP
153
154 ;SEE IF 'PIP' - 0, AND 'ATA', 'MOL' AND 'VV' 1
155 044740 013746 001350 8$: MOV $RMDSI,-(SP)
156 044744 042716 047677 BIC #^C<ATA.PIP.MOL.VV>,(SP)
157 044750 022726 110100 CMP #ATA.MOL!VV,(SP)+
158 044754 001002 BNE 9$ ;ERROR IN RMDS
159 044756 000137 045312 JMP 14$ ;RMDS IS OK
160
161 ;REPORT ERROR IF MOL - 0 AND OPI - 0
162 044762 032737 010000 001350 9$: BIT #MOL,$RMDSI ;IS MOL RESET??
163 044770 001030 BNE 10$ ;NO - MOL IS SET
164 044772 032737 020000 001352 BIT #OPI,$RMER1I ;WAS OPI SET
165 045000 001024 BNE 10$ ;YES - DONT REPORT ERROR
166 045002 013737 001350 001140 MOV $RMDSI,$GDDAT ;EXPECTED STATUS
167 045010 052737 010000 001140 BIS #MOL,$GDDAT
168 045016 013737 001350 001142 MOV $RMDSI,$BDDAT ;RECEIVED STATUS
169 045024 062716 000004 ADD #4,(SP)
170 045030 112776 000062 000000 MOVB #62,@(SP)
171 045036 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
  
```



```

172 045042 004736      JSR      PC,@(SP)+      ;REPORT ERROR VIA USER
173 045044 162716 000010 SUB      #10,(SP)
174 045050 000240      NOP
175
176
177 045052 032737 020000 001350 ;REPORT AN ERROR IF 'PIP' IS STIL SET AND SKI NOT SET
10$: BIT      #PIP,RMDSI      ;IS 'PIP' STILL SET??
178 045060 001430      BEQ      11$              ;NO!!
179 045062 032737 040000 001400 BIT      #SKI,RMER2I      ;WAS 'SKI' SET??
180 045070 001024      BNE      11$              ;YES-DONT REPORT PIP
181 045072 013737 001350 001140 MOV      RMDSI,$GDDAT     ;EXPECTED STATUS
182 045100 042737 020000 001142 BIC      #PIP,$BDDAT
183 045106 013737 001350 001142 MOV      RMDSI,$BDDAT     ;RECEIVED STATUS
184 045114 062716 000004      ADD      #4,(SP)          ;MOVE (SP) TO ERROR
185 045120 112776 000056 000000 MOVVB    #56,@(SP)        ;LOAD ERROR NUMBER
186 045126 162716 000002      SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
187 045132 004736      JSR      PC,@(SP)+      ;REPORT 'PIP' SET AFTER SEEK
188 045134 162716 000010 SUB      #10,(SP)          ;RESTORE (SP)
189 045140 000240      NOP
190
191
192 045142 032737 100000 001350 ;REPORT AN ERROR IF 'ATA' IS NOT SET
11$: BIT      #ATA,RMDSI      ;WAS 'ATA' SET ??
193 045150 001024      BNE      13$              ;YES!!
194 045152 013737 001350 001140 MOV      RMDSI,$GDDAT     ;EXPECTED STATUS
195 045160 052737 110600 001140 BIS      #ATA,MOL!DPR!DRY,$GDDAT
196 045166 013737 001350 001142 MOV      RMDSI,$BDDAT     ;RECEIVED STATUS
197 045174 062716 000004      ADD      #4,(SP)          ;MOVE (SP) TO ERROR
198 045200 112776 000057 000000 MOVVB    #57,@(SP)        ;LOAD ERROR NUMBER
199 045206 162716 000002      SUB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
200 045212 004736      JSR      PC,@(SP)+      ;REPORT ATTENTION NOT SET DURING
201
202 045214 162716 000010 SUB      #10,(SP)          ;SEEK TEST
203 045220 000240      NOP
204
205
206 045222 032737 000100 001350 ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
13$: BIT      #VV,RMDSI      ;IS VV = 0 ??
207 045230 001030      BNE      14$              ;NO..
208 045232 032737 010000 001400 BIT      #IVC,RMER2I      ;IS IVC ALSO 0 ??
209 045240 001024      BNE      14$              ;NO - IVC IS SET
210 045242 013737 001350 001140 MOV      RMDSI,$GDDAT     ;EXPECTED STATUS
211 045250 052737 000100 001140 BIS      #VV,$GDDAT
212 045256 013737 001350 001142 MOV      RMDSI,$BDDAT     ;RECEIVED STATUS
213 045264 062716 000004      ADD      #4,(SP)          ;ERROR #64
214 045270 112776 000064 000000 MOVVB    #64,@(SP)        ;MOVE SP TO RETURN FOR ERROR
215 045276 162716 000002      SUB      #2,(SP)
216 045302 004736      JSR      PC,@(SP)+
217 045304 162716 000010 SUB      #10,(SP)
218 045310 000240      NOP
219 045312
220
221
222 045312 000240      ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
223 045314 062716 000004      NOP
224 045320 105776 000000      ADD      #4,(SP)          ;MOVE (SP) TO ERROR CALL
225 045324 001403      TSTB    @(SP)            ;WAS ERROR CALLED??
226 045326 062716 000004      BEQ      15$              ;NO..
227 045332 000402      ADD      #4,(SP)          ;MOVE TO ERROR RETURN
228

```

229 045334 162716 000004
230 045340 000207
231
232 045342 000000

158: SUB #4.(SP)
168: RTS PC
3008: .WORD 0

:MOVE (SP) TO NO ERROR RETURN
:RETURN
:ERROR FLAGS

1
2
3
4
5
6
7
8
9
10
11

.SBTTL CONTROLLER CLEAR SUBROUTINE

: THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
: AND DRIVES, THEN SELECTS THE DRIVE.

```

:CALL: JSR      PC,CNTCLR
:       BR      ???
:       NOP
:       ERROR
:       ???
    
```

RETURN HERE IF NO ERROR FOUND
RETURN HERE IF ANY ERROR FOUND
SUB DEFINES ERROR NUMBER

12
13
14
15
16
17
18
19
20
21
22
23
24
25
26

```

045344
045344 010046
045346 010146
045350 013746 000004
045354 013746 000006
045360 012737 045420 000004
045366 012737 000300 000006
045374 013700 001276
045400 012760 000040 000010
045406 013701 001466
045412 111160 000010
045416 000412
045420 022626
045422 062766 000004 000010
045430 112776 000007 000010
045436 162766 000002 000010
045444
045444 012637 000006
045450 012637 000004
045454 012601
045456 012600
045460 000207
    
```

```

CNTCLR:
MOV      R0,-(SP)      ;;PUSH R0 ON STACK
MOV      R1,-(SP)      ;;PUSH R1 ON STACK
MOV      ERRVEC,-(SP)  ;;PUSH ERRVEC ON STACK
MOV      ERRVEC+2,-(SP) ;;PUSH ERRVEC+2 ON STACK
MOV      #10$,ERRVEC   ;SETUP FOR BUS TIMEOUT
MOV      #PR6,ERRVEC+2
MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
MOV      #CLR,RMCS2(R0) ;CLEAR MASSBUS
MOV      TSTQUE,R1     ;GET DEVICE UNDER TEST
MOVB     (R1),RMCS2(R0) ;SELECT DEVICE
BR       20$

10$:    CMP      (SP)+,(SP)+ ;ADJUST STACK
        ADD     #4,10(SP)   ;MOVE SP TO USER'S ERROR CALL
        MOVB   #7,@10(SP)  ;WRITE THE ERROR NUMBER
        SUB     #2,10(SP)   ;ADJUST SP TO RETURN TO ERROR

20$:    MOV      (SP)+,ERRVEC+2 ;;POP STACK INTO ERRVEC+2
        MOV      (SP)+,ERRVEC  ;;POP STACK INTO ERRVEC
        MOV      (SP)+,R1     ;;POP STACK INTO R1
        MOV      (SP)+,R0     ;;POP STACK INTO R0
RTS     PC
    
```

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

.SBTTL STATUS CHECK SUBROUTINES
:*****
.SBTTL CONTROLLER CLEAR STATUS CHECK SUBROUTINE

:THIS SUBROUTINE VERIFIES THAT THE SUBSYSTEM IS INITIALIZED BASED ON
:STATUS STORED IN THE GET BUFFER. THIS SUBROUTINE SHOULD ONLY BE
:USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
:5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.

:STATUS PERTINENT TO THE DEVICE IS NOT CHECKED. IN PARTICULAR, THE
:FOLLOWING STATUS BITS ARE NOT CHECKED:

: ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC

:CALL:
:(1) JSR PC,CLRSTS
: BR ??? RETURN HERE IF NO ERROR
: NOP RETURN HERE TO REPORT AN ERROR
: ERROR ERROR NUMBER DEFINED BY SUB
: JSR PC,@(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
: ??? RETURN HERE IF NO MORE ERRORS

CLRSTS:

:CLEAR USER'S ERROR CALL
ADD #4,(SP) ;MOVE SP TO ERROR
CLRB @(SP) ;CLEAR ERROR NUMBER
SUB #4,(SP) ;MOVE SP BACK TO NO ERROR

:REPORT ERROR IF RMCS1 NOT INITIALIZED
4\$: MOV RMCS1I,\$BDDAT ;VERIFY RMCS1
BIC #SC,\$BDDAT ;IGNORE SPECIAL CONDITION
MOV #DVA!RDY,\$GDDAT ;EXPECT DVA & RDY
CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED, RECEIVED
BEQ 5\$;BRANCH IF EQUAL
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #126,@(SP) ;WRITE ERROR NUMBER IN CALL
SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ ;REPORT ERROR VIA USER
SUB #10,(SP) ;MOVE SP BACK TO NO ERROR
NOP

:REPORT ERROR IF RMBA NOT RESET
5\$: CLR \$GDDAT ;VERIFY RMBA IS ZERO
MOV RMBAI,\$BDDAT
BEQ 7\$;BRANCH IF ZERO
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #127,@(SP) ;WRITE ERROR NUMBER IN CALL
SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ ;REPORT ERROR VIA USER
SUB #10,(SP) ;MOVE SP BACK TO NO ERROR
NOP

:REPORT ERROR IF RMCS2 NOT INITIALIZED
7\$: MOV RMCS2I,\$BDDAT ;VERIFY RMCS2
MOV R1,-(SP) ;PUSH R1 ON STACK
CLR -(SP) ;EXPECT IR & UNIT NUMBER
MOV TSTQUE,R1 ;R1 ADDRESS OF TEST QUE
MOVB (R1),(SP)

045462
045462 062716 000004
045466 105076 000000
045472 162716 000004
045476 013737 001336 001142
045504 042737 100000 001142
045512 012737 004200 001140
045520 023737 001140 001142
045526 001413
045530 062716 000004
045534 112776 000126 000000
045542 162716 000002
045546 004736
045550 162716 000010
045554 000240
045556 005037 001140
045562 013737 001342 001142
045570 001413
045572 062716 000004
045576 112776 000127 000000
045604 162716 000002
045610 004736
045612 162716 000010
045616 000240
045620 013737 001346 001142
045626 010146
045630 005046
045632 013701 001466
045636 111116

```

58 045640 052716 000100      BIS      #1R,(SP)
59 045644 012637 001140      MOV      (SP)+,$GDDAT
60 045650 012601              MOV      (SP)+,R1          ;;PGP STACK INTO R1
61 045652 023737 001140 001142  CMP      $GDDAT,$BDDAT    ;;COMPARE EXPECTED & RECEIVED
62 045660 001413              BEQ      9$              ;;BRANCH IF EQUAL
63 045662 062716 000004      ADD      #4,(SP)         ;;MOVE SP TO USER'S ERROR CALL
64 045666 112776 000130 000000  MOVVB   #130,@(SP)       ;;WRITE ERROR NUMBER IN CALL
65 045674 162716 000002      SUB      #2,(SP)         ;;MOVE SP TO RETURN FOR ERROR
66 045700 004736              JSR      PC,@(SP)+       ;;REPORT ERROR VIA USER
67 045702 162716 000010      SUB      #10,(SP)        ;;MOVE SP BACK TO NO ERROR
68 045706 000240              NOP
69                               ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
70 045710 005037 001140 9$:      CLR      $GDDAT          ;;VERIFY RMER1
71 045714 013737 001352 001142  MOV      RMER1I,$BDDAT
72 045722 042737 040000 001142  BIC      #UNS,$BDDAT     ;;IGNORE UNSAFE
73 045730 001413              BEQ      13$            ;;BRANCH IF ZERO
74 045732 062716 000004      ADD      #4,(SP)         ;;MOVE SP TO USER'S ERROR CALL
75 045736 112776 000131 000000  MOVVB   #131,@(SP)       ;;WRITE ERROR NUMBER IN CALL
76 045744 162716 000002      SUB      #2,(SP)         ;;MOVE SP TO RETURN FOR ERROR
77 045750 004736              JSR      PC,@(SP)+       ;;REPORT ERROR VIA USER
78 045752 162716 000010      SUB      #10,(SP)        ;;MOVE SP BACK TO NO ERROR
79 045756 000240              NOP
80                               ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
81 045760 013737 001362 001142 13$:   MOV      RMMR1I,$BDDAT   ;;VERIFY RMMR
82 045766 042737 000046 001142  BIC      #WC!LS!LST,$BDDAT ;;IGNORE WORD CLOCK, SCT, TRK
83 045774 012737 000010 001140  MOV      #MWD,$GDDAT     ;;EXPECT WRITE DATA BIT
84 046002 023737 001140 001142  CMP      $GDDAT,$BDDAT   ;;COMPARE EXPECTED AND RECEIVED
85 046010 001413              BEQ      17$            ;;BRANCH IF 0
86 046012 062716 000004      ADD      #4,(SP)         ;;MOVE SP TO USER'S ERROR CALL
87 046016 112776 000133 000000  MOVVB   #133,@(SP)       ;;WRITE ERROR NUMBER IN CALL
88 046024 162716 000002      SUB      #2,(SP)         ;;MOVE SP TO RETURN FOR ERROR
89 046030 004736              JSR      PC,@(SP)+       ;;REPORT ERROR VIA USER
90 046032 162716 000010      SUB      #10,(SP)        ;;MOVE SP BACK TO NO ERROR
91 046036 000240              NOP
92                               ;REPORT AN ERROR IF RMEC2 IS NOT RESET
93 046040 005037 001140 17$:   CLR      $GDDAT          ;;EXPECT ZEROS
94 046044 013737 001404 001142  MOV      RMEC2I,$BDDAT   ;;VERIFY RMEC2 0
95 046052 001413              BEQ      19$
96 046054 062716 000004      ADD      #4,(SP)         ;;MOVE SP TO USER'S ERROR CALL
97 046060 112776 000135 000000  MOVVB   #135,@(SP)       ;;WRITE ERROR NUMBER IN CALL
98 046066 162716 000002      SUB      #2,(SP)         ;;MOVE SP TO RETURN FOR ERROR
99 046072 004736              JSR      PC,@(SP)+       ;;REPORT ERROR VIA USER
100 046074 162716 000010     SUB      #10,(SP)        ;;MOVE SP BACK TO NO ERROR
101 046100 000240              NOP
102                               ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
103 046102 013737 001376 001142 19$:   MOV      RMMR2I,$BDDAT   ;;VERIFY RMMR2
104 046110 042737 140000 001142  BIC      #RQA!RQB,$BDDAT
105 046116 012737 011717 001140  MOV      #TST!1777,$GDDAT ;;EXPECT TEST BIT ON
106 046124 023737 001140 001142  CMP      $GDDAT,$BDDAT
107 046132 001413              BEQ      21$
108 046134 062716 000004      ADD      #4,(SP)         ;;MOVE SP TO USER'S ERROR CALL
109 046140 112776 000136 000000  MOVVB   #136,@(SP)       ;;WRITE ERROR NUMBER IN CALL
110 046146 162716 000002      SUB      #2,(SP)         ;;MOVE SP TO RETURN FOR ERROR
111 046152 004736              JSR      PC,@(SP)+       ;;REPORT ERROR VIA USER
112 046154 162716 000010     SUB      #10,(SP)        ;;MOVE SP BACK TO NO ERROR
113 046160 000240              NOP
114                               ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
    
```

```

115 046162 005037 001140 21$: CLR $GDDAT ;EXPECT ALL ZEROS
116 046166 013737 001400 001142 MOV RMER2I,$BDDAT ;VERIFY RMER2
117 046174 042737 040200 001142 BIC #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
118 046202 001413 BEQ 215$ ;BRANCH IF OTHER BITS 0
119 046204 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
120 046210 112776 000174 000000 MOVB #174,@(SP) ;WRITE ERROR NUMBER IN CALL
121 046216 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
122 046222 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
123 046224 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO NO ERROR
124 046230 000240 NOP
125 ;REPORT ERROR IF RMD5 NOT INITIALIZED
126 046232 013737 001350 001142 215$: MOV RMD5I,$BDDAT ;TEST DRIVE STATUS REGISTER
127 046240 042737 177177 001142 BIC #^C<DRY!DPR>,$BDDAT
128 046246 012737 000600 001140 MOV #DPR!DRY,$GDDAT ;EXPECTED DRIVE STATUS
129 046254 023737 001140 001142 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED & RECEIVED
130 046262 001413 BEQ 22$ ;BRANCH IF EQUAL
131 046264 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
132 046270 112776 000134 000000 MOVB #134,@(SP) ;WRITE ERROR NUMBER
133 046276 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
134 046302 004736 JSR PC,@(SP)+ ;REPORT ERROR TO USER
135 046304 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO NO ERROR
136 046310 000240 NOP
137 046312 062716 000004 22$: ADD #4,(SP) ;MOVE SP TO ERROR CALL
138 046316 105776 000000 TSTB @(SP) ;WAS AN ERROE DETECTED??
139 046322 001403 BEQ 23$ ;NO!!
140 046324 062716 000004 ADD #4,(SP) ;YES - MOVE TO ERROR RETURN
141 046330 000402 BR 24$
142 046332 162716 000004 23$: SUB #4,(SP) ;MOVE SP TO NO FR&OR RETURN
143 046336 000240 24$: NOP
144 046340 000207 RTS PC

```

.SBTTL PACK ACKNOWLEDGE STATUS CHECK

: THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
: COMMAND USING THE STATUS STORED IN THE GET BUFFER. ERRORS ARE
: REPORTED TO THE USER VIA THE USER'S ERROR CALL.

: CALL:
: (1) JSR PC,ACKSTS
: BR ??? RETURN HERE IF NO ERROR
: NOP RETURN HERE TO REPORT AN ERROR
: ERROR ERROR NUMBER DEFINED BY SUB
: JSR PC,@(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
: ??? RETURN HERE IF NO MORE ERRORS

ACKSTS:

: CLEAR USER'S ERROR CALL
ADD #4,(SP) : MOVE SP TO ERROR CALL
CLRB @(SP) : CLEAR LOW ORDER BYTE
SUB #4,(SP) : MOVE SP BACK

: REPORT AN ERROR IF 'VV' IS 0
BIT #VV,RMDSI : IS VOLUME VALID SET??
BNE 1\$: YES!!
MOV RMDSI,\$GDDAT : EXPECTED STATUS
BIS #VV,\$GDDAT
MOV RMDSI,\$BDDAT : RECEIVED STATUS
ADD #4,(SP) : MOVE SP TO ERROR CALL
MOVB #155,@(SP) : WRITE NUMBER IN ERROR CALL
SUB #2,(SP) : MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ : REPORT THE ERROR
SUB #10,(SP) : MOVE SP BACK TO BRANCH
NOP

1\$:

: REPORT AN ERROR IF 'MOL' IS 0
BIT #MOL,RMDSI : IS MOL SET??
BNE 2\$: YES!!
MOV RMDSI,\$GDDAT : EXPECTED STATUS
BIS #MOL,\$GDDAT
MOV RMDSI,\$BDDAT : RECEIVED STATUS
ADD #4,(SP) : MOVE SP TO ERROR CALL
MOVB #41,@(SP) : WRITE NUMBER OF ERROR IN CALL
SUB #2,(SP) : MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ : REPORT THE ERROR
SUB #10,(SP) : MOVE SP TO BRANCH
NOP

2\$:

: SEE IF 'UNS','OPI','RMR','ILR', OR 'ILF' IS SET
BIT #UNS:OPI:RMR:ILR:ILF,RMER1I
BEQ 7\$

: REPORT AN ERROR IF 'UNS' IS SET
BIT #UNS,RMER1I : WAS UNS SET??
BEQ 3\$: NO!
MOV RMER1I,\$BDDAT : RECEIVED STATUS

3
4
5
6
7
8
9
10
11
12
13
14
15 046342
16
17
18 046342 062716 000004
19 046346 105076 000000
20 046352 162716 000004
21
22
23 046356 032737 000100 001350
24 046364 001024
25 046366 013737 001350 001140
26 046374 052737 000100 001140
27 046402 013737 001350 001142
28 046410 062716 000004
29 046414 112776 000155 000000
30 046422 162716 000002
31 046426 004736
32 046430 162716 000010
33 046434 000240
34 046436
35
36
37 046436 032737 010000 001350
38 046444 001024
39 046446 013737 001350 001140
40 046454 052737 010000 001140
41 046462 013737 001350 001142
42 046470 062716 000004
43 046474 112776 000041 000000
44 046502 162716 000002
45 046506 004736
46 046510 162716 000010
47 046514 000240
48 046516
49
50
51 046516 032737 060007 001352
52 046524 001570
53
54
55 046526 032737 040000 001352
56 046534 001424
57 046536 013737 001352 001142

```

58 046544 013737 001352 001140      MOV      RMER11,$GDDAT      ;EXPECTED STATUS
59 046552 042737 040000 001140      BIC      #UNS,$GDDAT
60 046560 062716 000004      ADD      #4,(SP)          ;MOVE SP TO ERROR CALL
61 046564 112776 000042 000000      MOV      #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
62 046572 162716 000002      SUB      #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
63 046576 004736      JSR      PC,@(SP)+       ;REPORT THE ERROR VIA USER
64 046600 162716 000010      SUB      #10,(SP)        ;MOVE SP TO NO ERROR RETURN
65 046604 000240      NOP
66 046606      3$:
67
68      ;REPORT ANY OPI EPROR
69 046606 032737 020000 001352      BIT      #OPI,RMER11     ;WAS OPI SET ??
70 046614 001424      BEQ      4$              ;NO!!
71 046616 013737 001352 001142      MOV      RMER11,$BDDAT   ;RECEIVED STATUS
72 046624 013737 001352 001140      MOV      RMER11,$GDDAT   ;EXPECTED STATUS
73 046632 042737 020000 001140      BIC      #OPI,$GDDAT
74 046640 062716 000004      ADD      #4,(SP)         ;MOVE SP TO ERROR CALL
75 046644 112776 000043 000000      MOV      #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
76 046652 162716 000002      SUB      #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
77 046656 004736      JSR      PC,@(SP)+       ;REPORT THE ERROR VIA USER
78 046660 162716 000010      SUB      #10,(SP)        ;MOVE SP TO NO ERROR RETURN
79 046664 000240      NOP
80 046666      4$:
81
82      ;REPORT ANY RMR ERROR
83 046666 032737 000004 001352      BIT      #RMR,RMER11     ;WAS RMR SET??
84 046674 001424      BEQ      5$              ;NO!!
85 046676 013737 001352 001142      MOV      RMER11,$BDDAT   ;RECEIVED STATUS
86 046704 013737 001352 001140      MOV      RMER11,$GDDAT   ;EXPECTED STATUS
87 046712 042737 000004 001140      BIC      #RMR,$GDDAT
88 046720 062716 000004      ADD      #4,(SP)         ;MOVE SP TO ERROR CALL
89 046724 112776 000044 000000      MOV      #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
90 046732 162716 000002      SUB      #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
91 046736 004736      JSR      PC,@(SP)+       ;REPORT THE ERROR VIA USER
92 046740 162716 000010      SUB      #10,(SP)        ;MOVE SP TO NO ERROR RETURN
93 046744 000240      NOP
94 046746      5$:
95
96      ;REPORT ANY ILR ERROR
97 046746 032737 000002 001352      BIT      #ILR,RMER11     ;WAS ILR SET??
98 046754 001424      BEQ      6$              ;NO!!
99 046756 013737 001352 001142      MOV      RMER11,$BDDAT   ;RECEIVED STATUS
100 046764 013737 001352 001140      MOV      RMER11,$GDDAT   ;EXPECTED STATUS
101 046772 042737 000002 001140      BIC      #ILR,$GDDAT
102 047000 062716 000004      ADD      #4,(SP)         ;MOVE SP TO ERROR CALL
103 047004 112776 000045 000000      MOV      #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
104 047012 162716 000002      SUB      #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
105 047016 004736      JSR      PC,@(SP)+       ;REPORT THE ERROR VIA USER
106 047020 162716 000010      SUB      #10,(SP)        ;MOVE SP TO NO ERROR RETURN
107 047024 000240      NOP
108 047026      6$:
109
110      ;REPORT ANY ILF ERROR
111 047026 032737 000001 001352      BIT      #ILF,RMER11     ;WAS ILF SET??
112 047034 001424      BEQ      7$              ;NO!!
113 047036 013737 001352 001142      MOV      RMER11,$BDDAT   ;RECEIVED STATUS
114 047044 013737 001352 001140      MOV      RMER11,$GDDAT   ;EXPECTED STATUS

```


ZRMNBO RM05/3/2 FCTNL TST 2
PACK ACKNOWLEDGE STATUS CHECK

MACRO V04.0 4-APR-81 11:43:33 PAGE 32-2

SEQ 0199

115	047052	042737	000001	001140	BIC	#1LF,\$GDDAT	
116	047060	062716	C00004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
117	047064	112776	000046	000000	MOVB	#46,@(SP)	;WRITE NUMBER OF ERROR IN CALL
118	047072	162716	000002		SUB	#2,(SP)	;MOVE SP TO RETURN FOR ERROR
119	047076	004736			JSR	PC,@(SP)+	;REPORT THE ERROR VIA USER
120	047100	162716	000010		SUB	#10,(SP)	;MOVE SP TO NO ERROR RETURN
121	047104	000240			NOP		
122	047106						
123							
124							
125	047106	062716	000004		ADD	#4,(SP)	;MOVE SP TO ERROR CALL
126	047112	105776	000000		TSTB	@(SP)	;WAS ERROR FOUND??
127	047116	001403			BEQ	8\$;NO!
128	047120	062716	000004		ADD	#4,(SP)	;YES - MOVE TO ERROR RETURN
129	047124	000402			BR	9\$	
130	047126	162716	000004		SUB	#4,(SP)	;MOVE SP TO NO ERROR RETURN
131	047132	C00240			NOP		
132	047134	000207			RTS	PC	

7\$:

;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND

8\$:

9\$:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15 047136
16
17
18 047136 062716 000004
19 047142 105076 000000
20 047146 162716 000004
21
22
23
24 047152 032737 022011 001352
25 047160 001553
26
27
28
29 047162 032737 000010 001352
30 047170 001430
31 047172 032737 000010 001400
32 047200 001024
33 047202 013737 001352 001140
34 047210 042737 000010 001140
35 047216 013737 001352 001142
36 047224 062716 000004
37 047230 112776 000050 000000
38 047236 162716 000002
39 047242 004736
40 047244 162716 000010
41 047250 000240
42 047252
43
44
45 047252 032737 000001 001352
46 047260 001424
47 047262 013737 001352 001140
48 047270 042737 000001 001140
49 047276 013737 001352 001142
50 047304 062716 000004
51 047310 112776 000071 000000
52 047316 162716 000002
53 047322 004736
54 047324 162716 000010
55 047330 000240
56 047332
57

```

.SBTTL RECALIBRATE STATUS CHECK SUBROUTINE
;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
;USING THE STATUS STORED IN THE GET BUFFER.

;CALL:
;(1) JSR PC,RCLSTS ;CALL SUBROUTINE
      BR   ???      ;RETURN HERE IF NO ERROR
      NOP          ;RETURN HERE TO REPORT AN ERROR
      ERROR       ;ERROR NUMBER DEFINED BY SUB
      JSR PC,@(SP)+ ;GO BACK TO SUB FOR MORE ERROR CHECKS
      ???        ;RETURN HERE IF NO MORE ERRORS

RCLSTS:
;CLEAR USER'S ERROR NUMBER
ADD #4,(SP)
CLRB @(SP) ;CLEAR USER'S ERROR CALL
SUB #4,(SP) ;MOVE SP BACK TO BRANCH

;SEE IF 'PAR' OR 'ILF' OR 'OPI' OR 'IAE' IS SET
BIT #OPI.PAR.ILF.IAE,RMER1I
BEQ 4$ ;NONE ARE SET - GO TO NEXT CHECK

;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
;'PAR' - 1 AND 'DPE' = 0
BIT #PAR,RMER1I ;WAS 'PAR' SET??
BEQ 1$ ;NO!!
BIT #DPE,RMER2I ;WAS 'DPE' SET??
BNE 1$ ;YES - NOT A REGISTER ERROR
MOV RMER1I,$GDDAT ;EXPECTED STATUS
BIC #PAR,$GDDAT
MOV RMER1I,$BDDAT ;RECEIVED STATUS
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #50,@(SP) ;WRITE ERROR NUMBER IN CALL
SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ ;GO REPORT ERROR
SUB #10,(SP) ;MOVE SP BACK TO BRANCH
NOP

1$:
;REPORT ANY 'ILF' ERROR
BIT #ILF,RMER1I ;WAS 'ILF' SET??
BEQ 2$ ;NO!!
MOV RMER1I,$GDDAT ;EXPECTED STATUS
BIC #ILF,$GDDAT
MOV RMER1I,$BDDAT ;RECEIVED STATUS
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #71,@(SP) ;WRITE ERROR NUMBER IN CALL
SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ ;REPORT ERROR VIA USER
SUB #10,(SP) ;MOVE SP BACK TO BRANCH
NOP

2$:

```

```

58 ;REPORT ANY 'OPI' ERROR AS
59 :
60 : . OPI DUE TO 'MOL' = 0
61 : . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
61 047332 032737 020000 001352 BIT #OPI,RMER11 ;WAS OPI SET??
62 047340 001433 BEQ 31$ ;NO!!
63 047342 013737 001352 001140 MOV RMER11,$GDDAT ;EXPECTED STATUS
64 047350 042737 020000 001140 BIC #OPI,$GDDAT
65 047356 013737 001352 001142 MOV RMER11,$BDDAT ;RECEIVED STATUS
66 047364 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
67 047370 112776 000072 000000 MOVB #72,@(SP) ;WRITE ERROR NUMBER IN USER'S CALL
68 047376 032737 010000 001350 BIT #MOL,RMDSI ;WAS 'MOL' = 0??
69 047404 001403 BEQ 3$ ;YES!!
70 047406 112776 000073 000000 MOVB #73,@(SP) ;NO - CHANGE ERROR NUMBER
71 047414 162716 000002 3$: SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
72 047420 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
73 047422 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO BRANCH
74 047426 000240 NOP
75 047430 31$:
76
77 ;REPORT AN ERROR IF 'IAE' IS SET
78 047430 032737 002000 001352 BIT #IAE,RMER11 ;IS 'IAE' SET??
79 047436 001424 BEQ 4$ ;NO!!
80 047440 013737 001352 001140 MOV RMER11,$GDDAT ;EXPECTED STATUS
81 047446 042737 002000 001140 BIC #IAE,$GDDAT
82 047454 013737 001352 001142 MOV RMER11,$BDDAT ;RECEIVED STATUS
83 047462 062716 000004 ADD #4,(SP) ;MOVE SP TO ERROR CALL
84 047466 112776 000070 000000 MOVB #70,@(SP) ;WRITE ERROR NUMBER IN USER'S CALL
85 047474 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
86 047500 004736 JSR PC,@(SP)+ ;REPORT ERROR
87 047502 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO NO ERROR RETURN
88 047506 000240 NOP
89 047510 4$:
90
91 ;SEE IF 'SKI' OR 'IVC' OR 'DVC' IS SET
92 047510 032737 050200 001400 BIT #SKI.IVC.DVC,RMER21
93 047516 001517 BEQ 8$ ;NONE OF THE BITS ARE SET
94
95
96 ;REPORT ANY 'IVC' ERROR AS
97 :
98 : . IVC WITH VV - 0
99 : . ERRONEOUS IVC ERROR
99 047520 032737 010000 001400 BIT #IVC,RMER21 ;WAS IVC SET??
100 047526 001433 BEQ 6$ ;NO!!
101 047530 013737 001400 001140 MOV RMER21,$GDDAT ;EXPECTED STATUS
102 047536 042737 010000 001140 BIC #IVC,$GDDAT
103 047544 013737 001400 001142 MOV RMER21,$BDDAT ;RECEIVED STATUS
104 047552 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
105 047556 112776 000074 000000 MOVB #74,@(SP) ;WRITE ERROR NUMBER IN CALL
106 047564 032737 000100 001350 BIT #VV,RMDSI ;WAS VV - 0??
107 047572 001403 BEQ 5$ ;YES!!
108 047574 112776 000075 000000 MOVB #75,@(SP) ;NO - CHANGE ERROR NUMBER
109 047602 162716 000002 5$: SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
110 047606 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
111 047610 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO BRANCH
112 047614 000240 NOP
113 047616 6$:
114

```

```

115 ;REPORT ANY 'SKI' ERROR
116 047616 032737 040000 001400 BIT #SKI,RMER2I ;WAS SKI SET??
117 047624 001424 BEQ 7$ ;NO!!
118 047626 013737 001400 001140 MOV RMER2I,$GDDAT ;EXPECTED STATUS
119 047634 042737 040000 001140 BIC #SKI,$GDDAT
120 047642 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
121 047650 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
122 047654 112776 000076 000000 MOVB #76,@(SP) ;WRITE ERROR NUMBER
123 047662 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
124 047666 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
125 047670 162716 000010 SUB #10,(SP) ;MOVE SP TO BRANCH
126 047674 000240 NOP
127 047676 7$:
128
129 ;REPORT ANY 'DVC' ERROR
130 047676 032737 000200 001400 BIT #DVC,RMER2I ;WAS 'DVC' SET??
131 047704 001424 BEQ 8$ ;NO!!
132 047706 013737 001400 001140 MOV RMER2I,$GDDAT ;EXPECTED STATUS
133 047714 042737 000200 001140 BIC #DVC,$GDDAT
134 047722 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
135 047730 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
136 047734 112776 000077 000000 MOVB #77,@(SP) ;WRITE ERROR NUMBER
137 047742 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
138 047746 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
139 047750 162716 000010 SUB #10,(SP) ;MOVE SP TO USER'S BRANCH
140 047754 000240 NOP
141 047756 8$:
142
143 ;SEE IF 'PIP' AND 'OM' ARE 0, AND 'ATA','MOL' AND 'VV' ARE 1
144 047756 013746 001350 MOV RMDSI,-(SP) ;PUT RMDI ON STACK
145 047762 042716 047676 BIC #^C<PIP!MOL!VV.OM.ATA>,(SP)
146 047766 022726 110100 CMP #ATA!MOL.VV,(SP)+
147 047772 001002 BNE 85$
148 047774 000137 050410 JMP 13$
149 050000 85$:
150
151 ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
152 ;LINE AFTER RECALIBRATE WAS INITIATED
153 050000 032737 010000 001350 BIT #MOL,RMDSI ;DID MOL DROP??
154 050006 001030 BNE 9$ ;NO!!
155 050010 032737 020000 001352 BIT #OPI,RMER1I ;WAS OPI ERROR REPORTED??
156 050016 001024 BNE 9$ ;YES - DON'T REPORT MOL=0
157 050020 013737 001350 001140 MOV RMDSI,$GDDAT ;EXPECTED STATUS
158 050026 052737 010000 001140 BIS #MOL,$GDDAT
159 050034 013737 001350 001142 MOV RMDSI,$BDDAT ;RECEIVED STATUS
160 050042 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
161 050046 112776 000100 000000 MOVB #100,@(SP) ;WRITE ERROR NUMBER
162 050054 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
163 050060 004736 JSR PC,@(SP)+ ;REPORT ERROR VIA USER
164 050062 162716 000010 SUB #10,(SP) ;MOVE SP BACK TO USER'S BRANCH
165 050066 000240 NOP
166 050070 9$:
167
168 ;REPORT AN ERROR IF 'VV' = 0 AND 'IVC' = 0
169 050070 032737 000100 001350 BIT #VV,RMDSI ;DID 'VV' DROP??
170 050076 001030 BNE 10$ ;NO!!
171 050100 032737 010000 001400 BIT #IVC,RMER2I ;WAS THERE A IVC ERROR??

```

```

172 050 001024      BNE      10$      ;YES - DONT REPORT VV=0
173 050110 013737 001350 001140  MOV      RMDSI,$GDDAT ;EXPECTED STATUS
174 050116 013737 001350 001142  MOV      RMDSI,$BDDAT ;RECEIVED STATUS
175 050124 052737 000100 001140  BIS      #VV,$GDDAT
176 050132 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
177 050136 112776 000101 000000  MOVVB   #101,@(SP)   ;WRITE ERROR NUMBER IN CALL
178 050144 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
179 050150 004736      JSR      PC,@(SP)+
180 050152 162716 000010  SUB      #10,(SP)    ;MOVE SP BACK TO USER'S BRANCH
181 050156 000240      NOP
182 050160      10$:
183
184      ;REPORT AN ERROR IF ATA IS NOT SET
185 050160 032737 100000 001350  BIT      #ATA,RMDSI   ;WAS ATA SET DURING RECALIBRATE??
186 050166 001024      BNE      11$      ;YES!
187 050170 013737 001350 001140  MOV      RMDSI,$GDDAT ;EXPECTED STATUS
188 050176 052737 100000 001140  BIS      #ATA,$GDDAT
189 050204 013737 001350 001142  MOV      RMDSI,$BDDAT ;RECEIVED STATUS
190 050212 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
191 050216 112776 000102 000000  MOVVB   #102,@(SP)   ;WRITE ERROR NUMBER IN CALL
192 050224 162716 000002      SUB      #2,(SP)      ;MOVE SP TO USER'S BRANCH
193 050230 004736      JSR      PC,@(SP)+
194 050232 162716 000010  SUB      #10,(SP)
195 050236 000240      NOP
196
197 050240      11$:
198
199      ;REPORT AN ERROR IF 'OM' IS NOT ZERO BECAUSE RECALIBRATE SHOULD
200      ;ALWAYS CLEAR OFFSET MODE
201 050240 032737 000001 001350  BIT      #OM,RMDSI   ;WAS 'OM' RESET??
202 050246 001424      BEQ      12$      ;YES!
203 050250 013737 001350 001140  MOV      RMDSI,$GDDAT ;EXPECTED STATUS
204 050256 042737 000001 001140  BIC      #OM,$GDDAT
205 050264 013737 001350 001142  MOV      RMDSI,$BDDAT ;RECEIVED STATUS
206 050272 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
207 050276 112776 000103 000000  MOVVB   #103,@(SP)   ;WRITE ERROR NUMBER
208 050304 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
209 050310 004736      JSR      PC,@(SP)+   ;REPORT ERROR VIA USER
210 050312 162716 000010  SUB      #10,(SP)    ;MOVE SP TO USER'S BRANCH
211 050316 000240      NOP
212 050320      12$:
213
214      ;REPORT AN ERROR IF 'PIP' IS STIL ON, I.E., DRIVE NOT ON
215      ;CYLINDER
216 050320 032737 020000 001350  BIT      #PIP,RMDSI  ;IS DRIVE OFF CYLINDER??
217 050326 001430      BEQ      13$      ;NO!
218 050330 032737 040000 001400  BIT      #SKI,RMER2I ;WAS 'SKI' DETECTED??
219 050336 001024      BNE      13$      ;YES-DONT REPORT 'PIP'
220 050340 013737 001350 001140  MOV      RMDSI,$GDDAT ;EXPECTED STATUS
221 050346 042737 020000 001140  BIC      #PIP,$GDDAT
222 050354 013737 001350 001142  MOV      RMDSI,$BDDAT ;RECEIVED STATUS
223 050362 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR CALL
224 050366 112776 000104 000000  MOVVB   #104,@(SP)   ;WRITE ERROR NUMBER
225 050374 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
226 050400 004736      JSR      PC,@(SP)+
227 050402 162716 000010  SUB      #10,(SP)    ;MOVE SP BACK TO USER'S BRANCH
228 050406 000240      NOP
    
```

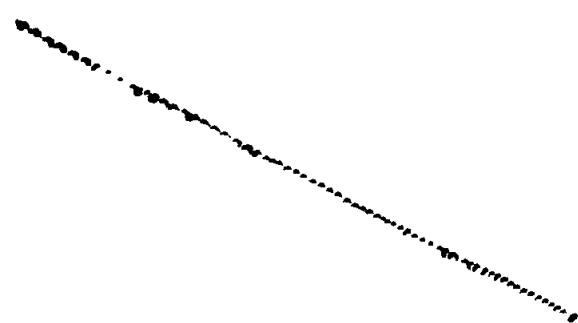
```

229 050410      13$:
230
231             ;SEE IF 'ILR' OR 'RMR' OR 'UNS' IS SET
232 050410 032737 040006 001352      BIT      #ILR!RMR!UNS,RMER1I
233 050416 001514             BEQ      16$
234
235             ;REPORT AN ERROR IF 'ILR' IS SET
236 050420 032737 000002 001352      BIT      #ILR,RMER1I      ;WAS ILR SET DURING RECALIBRATE??
237 050426 001424             BEQ      14$      ;NO!!
238 050430 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
239 050436 042737 000002 001140      BIC      #ILR,$GDDAT
240 050444 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
241 050452 062716 000004             ADD      #4,(SP) ;MOVE SP TO USER'S ERROR CALL
242 050456 112776 000105 000000      MOV      #105,@(SP) ;WRITE ERROR NUMBER IN CALL
243 050464 162716 000002             SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
244 050470 004736             JSR      PC,@(SP)+
245 050472 162716 000010             SUB      #10,(SP) ;MOVE SP TO USER'S BRANCH
246 050476 000240             NOP
247 050500
248
249             ;REPORT AN ERROR IF 'RMR' IS SET
250 050500 032737 000004 001352      BIT      #RMR,RMER1I      ;WAS RMR SET??
251 050506 001424             BEQ      15$      ;NO!!
252 050510 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
253 050516 042737 000004 001140      BIC      #RMR,$GDDAT
254 050524 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
255 050532 062716 000004             ADD      #4,(SP) ;MOVE SP TO USER'S ERROR CALL
256 050536 112776 000106 000000      MOV      #106,@(SP) ;WRITE ERROR NUMBER IN USER'S CALL
257 050544 162716 000002             SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
258 050550 004736             JSR      PC,@(SP)+ ;REPORT ERROR VIA USER
259 050552 162716 000010             SUB      #10,(SP) ;MOVE SP TO USER'S BRANCH
260 050556 000240             NOP
261 050560
262
263             ;REPORT AN ERROR IF 'UNS' IS SET AND 'DVC' IS 0
264 050560 032737 040000 001352      BIT      #UNS,RMER1I      ;WAS UNSAFE ON??
265 050566 001430             BEQ      16$      ;NO!!
266 050570 032737 000200 001400      BIT      #DVC,RMER2I      ;WAS THERE A DEVICE CHECK??
267 050576 001024             BNE      16$      ;YES - DON'T REPORT UNSAFE
268 050600 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
269 050606 042737 040000 001140      BIC      #UNS,$GDDAT
270 050614 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
271 050622 062716 000004             ADD      #4,(SP) ;MOVE SP TO USER'S ERROR CALL
272 050626 112776 000107 000000      MOV      #107,@(SP) ;WRITE ERROR NUMBER
273 050634 162716 000002             SUB      #2,(SP) ;MOVE SP TO RETURN FOR ERROR
274 050640 004736             JSR      PC,@(SP)+ ;REPORT ERROR VIA USER
275 050642 162716 000010             SUB      #10,(SP) ;MOVE SP BACK TO USER'S BRANCH
276 050646 000240             NOP
277 050650
278
279             ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
280 050650 062716 000004             ADD      #4,(SP) ;MOVE SP TO USER'S ERROR CALL
281 050654 105776 000000             TSTB    @(SP) ;WAS AN ERROR REPORTED??
282 050660 001403             BEQ      17$      ;NO..
283 050662 062716 000004             ADD      #4,(SP) ;YES - AUGMENT SP RETURN
284 050666 000402             BR      18$
285 050670 162716 000004             SUB      #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
    
```

286 050674 000240
287 050676 000207

185: NOP
RTS PC

;STATUS CECK IS COMPLETE



1
 2
 3
 4
 5
 6
 7
 8
 9 050700
 10
 11
 12 050700 062716 000004
 13 050704 105076 000000
 14 050710 162716 000004
 15
 16 050714 013737 001336 001142
 17 050722 042737 173700 001142
 18 050730 012737 004010 001140
 19 050736 023737 001140 001142
 20 050744 001443
 21 050746 062716 000004
 22 050752 112776 000141 000000
 23 050760 162716 000002
 24 050764 004736
 25 050766 162716 000010
 26 050772 000240
 27
 28 050774 013737 001350 001142
 29 051002 042737 021101 001142
 30 051010 012737 010600 001140
 31 051016 023737 001140 001142
 32 051024 001413
 33 051026 062716 000004
 34 051032 112776 000142 000000
 35 051040 162716 000002
 36 051044 004736
 37 051046 162716 000010
 38 051052 000240
 39
 40 051054 005037 001140
 41 051060 013737 001352 001142
 42 051066 001413
 43 051070 062716 000004
 44 051074 112776 000143 000000
 45 051102 162716 000002
 46 051106 004736
 47 051110 162716 000010
 48 051114 000240
 49
 50 051116 013737 001354 001142
 51 051124 010146
 52 051126 010246
 53 051130 013701 001466
 54 051134 116102 000001
 55 051140 042702 177400
 56 051144 005102
 57 051146 040237 001142

```

.SBTTL DRIVE CLEAR STATUS CHECK SUBROUTINE
:      BR      ???      RETURN HERE IF NO ERROR
:      NOP      RETURN HERE TO REPORT AN ERROR
:      ERROR   ERROR NUMBER DEFINED BY SUB
:      JSR     PC,@(SP)+ GO BACK TO SUB FOR MORE ERROR CHECKS
:      ???     RETURN HERE IF NO MORE ERRORS

DRVSTS:

: CLEAR USER'S ERROR CALL
      ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
      CLRB   @(SP)        ;CLEAR ERROR CALL
      SUB     #4,(SP)      ;MOVE SP TO USER'S BRANCH
: REPORT ERROR IF RMCS1 NOT INITIALIZED
4$:   MOV     RMCS1I,$BDDAT ;CHECK RMCS1
      BIC    #^C<DVA.FNCMSK>,$BDDAT ;CLEAR DONT CARES
      MOV     #DVA.DRVCLR,$GDDAT ;EXPECT DVA
      CMP     $GDDAT,$BDDAT ;COMPARE EXPECTED & RECEIVED
      BEQ     6$          ;BRANCH IF EQUAL
      ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
      MOVB   #141,@(SP)   ;WRITE NUMBER OF ERROR IN CALL
      SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
      JSR     PC,@(SP)+   ;REPORT THE ERROR VIA USER
      SUB     #10,(SP)     ;MOVE SP TO NO ERROR RETURN
      NOP

: REPORT ERROR IF RMD5 NOT INITIALIZED
5$:   MOV     RMD5I,$BDDAT ;CHECK RMD5
      BIC    #PGM!OM!VV!PIP,$BDDAT ;CLEAR DONT CARES
      MOV     #MOL!DPR!DRY,$GDDAT ;EXPECT DRY & DPR
      CMP     $GDDAT,$BDDAT ;COMPARE EXPECTED & RECEIVED
      BEQ     6$          ;BRANCH IF EQUAL
      ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
      MOVB   #142,@(SP)   ;WRITE NUMBER OF ERROR IN CALL
      SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
      JSR     PC,@(SP)+   ;REPORT THE ERROR VIA USER
      SUB     #10,(SP)     ;MOVE SP TO NO ERROR RETURN
      NOP

: REPORT ERROR IF RMER1 NOT INITIALIZED
6$:   CLR     $GDDAT       ;EXPECT 0'S
      MOV     RMER1I,$BDDAT ;CHECK RMER1
      BEQ     8$          ;BRANCH IF EQUAL
      ADD     #4,(SP)      ;MOVE SP TO ERROR CALL
      MOVB   #143,@(SP)   ;WRITE NUMBER OF ERROR IN CALL
      SUB     #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
      JSR     PC,@(SP)+   ;REPORT THE ERROR VIA USER
      SUB     #10,(SP)     ;MOVE SP TO NO ERROR RETURN
      NOP

: REPORT ERROR IF ATA NOT INITIALIZED
8$:   MOV     RMASI,$BDDAT ;CHECK ATTENTION BIT
      MOV     R1,-(SP)     ;;PUSH R1 ON STACK
      MOV     R2,-(SP)     ;;PUSH R2 ON STACK
      MOV     TSTQUE,R1
      MOVB   1(R1),R2
      BIC    #^CATNMSK,R2
      COM    R2
      BIC    R2,$BDDAT
  
```



```

58 051152 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
59 051154 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
60 051156 005737 001142  TST      $BDDAT      ;;IS ATTENTION CLEARED??
61 051162 001413      BEQ      9$           ;;BRANCH IF ATTENTION CLEARED
62 051164 062716 000004  ADD      #4,(SP)      ;;MOVE SP TO ERROR CALL
63 051170 112776 000144 000000  MOVB    #144,@(SP)    ;;WRITE NUMBER OF ERROR IN CALL
64 051176 162716 000002  SUB      #2,(SP)      ;;MOVE SP TO RETURN FOR ERROR
65 051202 004736      JSR      PC,@(SP)+    ;;REPORT THE ERROR VIA USER
66 051204 162716 000010  SUB      #10,(SP)     ;;MOVE SP TO NO ERROR RETURN
67 051210 000240      NOP
68
69 051212 013737 001362 001142 9$:      ;REPORT ERROR IF RMMR1 NOT INITIALIZED
70 051220 042737 000046 001142  MOV      RMMR1I,$BDDAT ;CHECK RMMR
71 051226 012737 000010 001140  BIC      #WC!LS!LST,$BDDAT ;CLEAR DONT CARES
72 051234 023737 001140 001142  MOV      #MWD,$GDDAT  ;EXPECT WRITE DATA ON
73 051242 001413      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED AND RECEIVED
74 051244 062716 000004  BEQ      11$          ;BRANCH IF ZERO
75 051250 112776 000145 000000  ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
76 051256 162716 000002  MOVB    #145,@(SP)    ;WRITE NUMBER OF ERROR IN CALL
77 051262 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
78 051264 162716 000010  JSR      PC,@(SP)+    ;REPORT THE ERROR VIA USER
79 051270 000240      SUB      #10,(SP)     ;MOVE SP TO NO ERROR RETURN
80      NOP
81 051272 013737 001376 001142 11$:     ;REPORT ERROR IF RMMR2 NOT INITIALIZED
82 051300 042737 140000 001142  MOV      RMMR2I,$BDDAT ;CHECK RMMR2
83 051306 012737 011777 001140  BIC      #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
84 051314 023737 001140 001142  MOV      #TST:1777,$GDDAT ;EXPECT TEST BIT ON
85 051322 001413      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED & RECEIVED
86 051324 062716 000004  BEQ      15$          ;BRANCH IF EQUAL
87 051330 112776 000146 000000  ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
88 051336 162716 000002  MOVB    #146,@(SP)    ;WRITE NUMBER OF ERROR IN CALL
89 051342 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
90 051344 162716 000010  JSR      PC,@(SP)+    ;REPORT THE ERROR VIA USER
91 051350 000240      SUB      #10,(SP)     ;MOVE SP TO NO ERROR RETURN
92 051352 005037 001140 15$:     CLR      $GDDAT      ;EXPECT ZEROS
93
94 051356 013737 001404 001142 17$:     ;REPORT ERROR IF RMEC2 NOT RESET
95 051364 001413      MOV      RMEC2I,$BDDAT ;CHECK RMEC2
96 051366 062716 000004  BEQ      17$          ;BRANCH IF 0
97 051372 112776 000150 000000  ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
98 051400 162716 000002  MOVB    #150,@(SP)    ;WRITE NUMBER OF ERROR IN CALL
99 051404 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
100 051406 162716 000010  JSR      PC,@(SP)+    ;REPORT THE ERROR VIA USER
101 051412 000240      SUB      #10,(SP)     ;MOVE SP TO NO ERROR RETURN
102      NOP
103 051414 013737 001400 001142 17$:     ;REPORT ERROR IF RMER2 NOT RESET
104 051422 001413      MOV      RMER2I,$BDDAT ;CHECK RMER2
105 051424 062716 000004  BEQ      18$          ;BRANCH IF NO ERROR
106 051430 112776 000147 000000  ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
107 051436 162716 000002  MOVB    #147,@(SP)    ;WRITE NUMBER OF ERROR IN CALL
108 051442 004736      SUB      #2,(SP)      ;MOVE SP TO RETURN FOR ERROR
109 051444 162716 000010  JSR      PC,@(SP)+    ;REPORT THE ERROR VIA USER
110 051450 000240      SUB      #10,(SP)     ;MOVE SP TO NO ERROR RETURN
111 051452      NOP
112
113 051452      18$:
114      19$:

```

```
115  
116 051452 062716 000004  
117 051456 105776 000000  
118 051462 001403  
119 051464 062716 000004  
120 051470 000402  
121 051472 162716 000004  
122 051476 000240  
123 051500 000207  
  
; AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND  
ADD #4, (SP) ; MOVE SP TO ERROR CALL  
TSTB @ (SP) ; WAS AN ERROR DETECTED??  
BEQ 21$ ; NO!!  
ADD #4, (SP) ; YES - MOVE SP TO ERROR RETURN  
BR 23$  
21$: SUB #4, (SP) ; MOVE SP BACK TO NO ERROR RETURN  
23$: NOP  
RTS PC ; RETURN TO USER
```

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

```

.SBTTL DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
;STORED IN THE PUT BUFFER. ERRORS ARE REPORTED BY WRITING
;THE ERROR NUMBER IN THE USERS ERROR CALL.

;USER'S SUBROUTINE CALL:
;(1) JSR PC,DTASTS
;(2) BR ??
;(3) NOP
;(4) ERROR
;(5) JSR PC,@(SP)+
;(6) ??
;

DTASTS:

;CLEAR USER'S ERROR CALL AND ERROR FLAGS
ADD #4,(SP) ;MOVE SP TO USER'S ERROR
CLRB @(SP) ;CLEAR LOW ORDER BYTE OF TRAP
SUB #4,(SP) ;RESTORE SP TO NO ERROR
CLR 500$ ;CLEAR ERROR FLAGS

;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS,
;I.E., MCPE = 1
BIT #MCPE,RMCS1I ;WAS THERE A PARITY ERROR??
BEQ 10$ ;NO!!
MOV RMCS1I,$GDDAT ;EXPECTED STATUS
BIC #MCPE,$GDDAT
MOV RMCS1I,$BDDAT ;RECEIVED STATUS
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #13,@(SP) ;WRITE ERROR NUMBER
SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
BR 30$

10$:

;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRITING REMOTE REGISTERS,
;I.E., PAR = 1 AND DPE = 0
BIT #PAR,RMER1I ;WAS THERE A PARITY ERROR??
BEQ 20$ ;NO!!
BIT #DPE,RMER2I ;DATA PARITY ERROR ?
BNE 20$ ;YES!!
MOV RMER1I,$GDDAT ;EXPECTED STATUS
BIC #PAR,$GDDAT
MOV RMER1I,$BDDAT ;RECEIVED STATUS
ADD #4,(SP) ;MOVE SP TO USER'S ERROR
MOVB #50,@(SP) ;WRITE ERROR NUMBER
BIT #MXF,RMCS2I ;DID MXF GET SET??
BNE 15$ ;YES!!
MOVB #274,@(SP) ;NO - CHANGE ERROR NUMBER
SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
BR 30$

;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
    
```

051502				
051502	062716	0C0004		
051506	105076	000000		
051512	162716	000004		
051516	005037	055076		
051522	032737	020000	001336	
051530	001422			
051532	013737	001336	001140	
051540	042737	020000	001140	
051546	013737	001336	001142	
051554	062716	000004		
051560	112776	000013	000000	
051566	162716	000002		
051572	004736			
051574	000466			
051576				
051576	032737	000010	001352	
051604	001435			
051606	032737	000010	001400	
051614	001031			
051616	013737	001352	001140	
051624	042737	000010	001140	
051632	013737	001352	001142	
051640	062716	000004		
051644	112776	000050	000000	
051652	032737	001000	001346	
051660	001003			
051662	112776	000274	000000	
051670	162716	000002		
051674	004736			
051676	000425			

```

58 ;MECHANICAL POSITIONING
59
60 ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
61 ;CODE AND GO BIT WERE LOADED
62 051700 20$:
63 051700 032737 001000 001346 BIT #MXF,RMCS2I ;WAS 'MISSED TRANSFER' SET??
64 051706 001425 BEQ 40$ ;NO!!
65 051710 013737 001346 001140 MOV RMCS2I,$GDDAT ;EXPECTED STATUS
66 051716 042737 001000 001140 BIC #MXF,$GDDAT
67 051724 013737 001346 001142 MOV RMCS2I,$BDDAT ;RECEIVED STATUS
68 051732 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
69 051736 112776 000275 000000 MOVB #275,@(SP) ;WRITE ERROR NUMBER
70 051744 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
71 051750 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
72 051752
73
74 ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
75 051752 162716 000010 SUB #10,(SP) ;MOVE SP TO NO ERROR
76 051756 000137 055050 JMP 380$ ;SKIP TO END OF SUB
77
78 40$:
79
80 ;REPORT AN ERROR IF 'OPI' ERROR OCCURRED DUE TO 'MOL' - 0, OR IF 'OPI'
81 ;AND 'MOL' ARE SET, BUT 'VV' IS RESET, INDICATING AN INTERMITTENT
82 ;'MOL'
83 051762 032737 020000 001352 BIT #OPI,RMER1I ;IS 'OPI' SET??
84 051770 001447 BEQ 60$ ;NO!!
85 051772 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
86 052000 042737 020000 001140 BIC #OPI,$GDDAT
87 052006 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
88 052014 032737 010000 001350 BIT #MOL,RMDSI ;WAS MEDIUM OFF LINE??
89 052022 001404 BEQ 45$ ;YES!!
90 052024 032737 000100 001350 BIT #VV,RMDSI ;WAS 'MOL' INTERMITTENT??
91 052032 001013 BNE 50$ ;NO!
92 052034 062716 000004 45$: ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
93 052040 112776 000276 000000 MOVB #276,@(SP) ;WRITE ERROR NUMBER IN CALL
94 052046 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
95 052052 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
96 052054 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
97 052060 000413 BR 60$
98 052062
99
100 ;REPORT 'OPI' ERROR, WHICH IS DUE TO 'ON CYLINDER' NOT DROPPING OR
101 ;'RUN' TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
102 052062 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
103 052066 112776 000277 000000 MOVB #277,@(SP) ;WRITE ERROR NUMBER IN CALL
104 052074 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
105 052100 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
106 052102 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
107 052106 000240 NOP
108 052110
109
110 ;LOOK FOR 'IVC' ERROR DURING COMMAND INITIATION
111 052110 032737 010000 001400 BIT #IVC,RMER2I ;WAS THERE AN 'IVC' ERROR??
112 052116 001432 BEQ 70$ ;NO!
113 ;REPORT 'IVC' ERROR DUE TO 'VV' 0, OR REPORT ERRONEOUS 'IVC' ERROR
114 052120 013737 001400 001140 MOV RMER2I,$GDDAT ;EXPECTED STATUS

```

```

115 052126 042737 010000 001140 BIC #IVC,$GDDAT
116 052134 013737 001400 001142 MOV RMER2I,$BDDAT ;RECEIVED STATUS
117 052142 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR
118 052146 112776 000300 000000 MOVB #300,@(SP) ;WRITE ERROR NUMBER IN CALL
119 052154 032737 000100 001350 BIT #VV,RMDSI ;WAS VOLUME VALID??
120 052162 001403 BEQ 65$ ;NO!!
121 052164 112776 000301 000000 MOVB #301,@(SP) ;CHANGE ERROR NUMBER
122 052172 162716 000002 65$: SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
123 052176 004736 JSR PC,@(SP)+ ;REPORT 'IVC' ERROR AND RETURN
124 052200 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
125 052204 70$:
126
127 ;SEE IF 'ILF' OR 'RMR' IS SET
128 052204 032737 000007 001352 BIT #ILR,ILF!RMR,RMER1I
129 052212 001510 BEQ 100$ ;NO ERRORS DETECTED
130 ;REPORT AN ERROR IF 'ILR' IS SET
131 052214 032737 000002 001352 BIT #ILR,RMER1I ;WAS 'ILR' DETECTED??
132 052222 001424 BEQ 80$ ;NO!
133 052224 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
134 052232 042737 000002 001140 BIC #ILR,$GDDAT
135 052240 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
136 052246 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
137 052252 112776 000302 000000 MOVB #302,@(SP) ;WRITE ERROR NUMBER IN CALL
138 052260 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
139 052264 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
140 052266 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
141 052272 000240 NOP
142 052274 80$:
143
144 ;REPORT AN ERROR IF 'ILF' IS SET
145 052274 032737 000001 001352 BIT #ILF,RMER1I ;WAS 'ILF' DETECTED??
146 052302 001424 BEQ 90$ ;NO!!
147 052304 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
148 052312 042737 000001 001140 BIC #ILF,$GDDAT
149 052320 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
150 052326 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
151 052332 112776 000303 000000 MOVB #303,@(SP) ;WRITE ERROR NUMBER IN CALL
152 052340 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
153 052344 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
154 052346 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
155 052352 000240 NOP
156 052354 90$:
157
158 ;REPORT AN ERROR IF 'RMR' IS SET
159 052354 032737 000004 001352 BIT #RMR,RMER1I ;WAS 'RMR' DETECTED??
160 052362 001424 BEQ 100$ ;NO!
161 052364 013737 001352 001140 MOV RMER1I,$GDDAT ;EXPECTED STATUS
162 052372 042737 000004 001140 BIC #RMR,$GDDAT
163 052400 013737 001352 001142 MOV RMER1I,$BDDAT ;RECEIVED STATUS
164 052406 062716 000004 ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
165 052412 112776 000304 000000 MOVB #304,@(SP) ;WRITE ERROR NUMBER IN CALL
166 052420 162716 000002 SUB #2,(SP) ;MOVE SP TO RETURN IF ERROR
167 052424 004736 JSR PC,@(SP)+ ;REPORT ERROR AND RETURN
168 052426 162716 000010 SUB #10,(SP) ;RESTORE SP TO NO ERROR
169 052432 000240 NOP
170 052434 100$:
171 ;DETERMINE WHETHER OR NOT 'IAE' SHOULD BE SET AND CHECK FOR ERROR
    
```

```

172 052434 012737 002000 001140      MOV      #IAE,$GDDAT      ;SETUP FOR 'IAE' - 1
173 052442 052737 040000 055076      BIS      #SKI,500$      ;SETUP FOR 'SKI' = 1
174 052450 023727 001446 001466      CMP      RMDCO,#822.    ;GREATER THAN LAST CYLINDER ?
175 052456 101025                BHI      110$          ;YES - CYLINDER IS INVALID
176 052460 042737 040000 055076      BIC      #SKI,500$      ;RESET SKI FLAG
177
178 052466 123737 001421 001335      CMPB     RMDAO+1,LSTRK+1 ;GREATER THAN LAST TRACK ?
179 052474 101016                BHI      110$          ;YES - TRACK IS INVALID
180
181 052476 123727 001420 000035      CMPB     RMDAO,#29.    ;IS SECTOR > 29. ?
182 052504 101410                BLOS     105$          ;NO
183 052506 032737 010000 001444      BIT      #FM16,RMOFO    ;18 BIT FORMAT ?
184 052514 001406                BEQ      110$          ;YES - SECTOR IS INVALID FOR 18 BIT MODE
185 052516 123727 001420 000037      CMPB     RMDAO,#31.    ;IS SECTOR > 31. ?
186 052524 101002                BHI      110$          ;YES - SECTOR IS INVALID
187 052526 005037 001140      105$:    CLR      $GDDAT        ;'IAE' SHOULD = 0
188
189 052532 013737 001352 001142      110$:    MOV      RMER1I,$BDDAT   ;GET RECEIVED STATUS
190 052540 042737 175777 001142      BIC      #^CIAE,$BDDAT
191 052546 023737 001140 001142      CMP      $GDDAT,$BDDAT ;IS 'IAE' STATUS OK??
192 052554 001004                BNE      115$          ;NO!!
193 052556 042737 040000 055076      BIC      #SKI,500$      ;IAE OK - SKI SHOULD BE 0
194 052564 000412                BR
195 052566 062716 000004      115$:    ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
196 052572 112776 000305 000000      MOVVB    #305,@(SP)     ;WRITE ERROR NUMBER
197 052600 162716 000002      SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
198 052604 004736                JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
199 052606 162716 000010      SUB      #10,(SP)       ;MOVE SP TO NO ERROR
200 052612
201
202
203 052612 013737 001400 001142      ;REPORT AN ERROR IF 'SKI' IS SET AND 'IAE' STATUS WAS OK
204 052620 042737 137777 001142      MOV      RMER2I,$BDDAT ;RECEIVED STATUS
205 052626 013737 055076 001140      BIC      #^CSKI,$BDDAT
206 052634 042737 137777 001140      MOV      500$,$GDDAT   ;EXPECTED STATUS
207 052642 032737 040000 001400      BIC      #^CSKI,$GDDAT
208 052650 001417                BIT      #SKI,RMER2I    ;WAS 'SKI' SET??
209 052652 032737 040000 055076      BEQ      140$          ;NO!!
210 052660 001032                BIT      #SKI,500$      ;WAS SKI CAUSED BY IAE 0??
211 052662 062716 000004      BNE      150$          ;YES - DON'T REPORT SKI
212 052666 112776 000306 000000      ADD      #4,(SP)        ;MOVE SP TO USERS ERROR CALL
213 052674 162716 000002      MOVVB    #306,@(SP)     ;WRITE ERROR NUMBER
214 052700 004736                SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
215 052702 162716 000010      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
216 052706 000417                SUB      #10,(SP)       ;MOVE SP TO NO ERROR
217
218 052710      140$:
219
220
221 052710 032737 040000 055076      ;REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED
222 052716 001413                BIT      #SKI,500$      ;SHOULD SKI BE SET??
223 052720 062716 000004      BEQ      150$          ;NO!!
224 052724 112776 000307 000000      ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
225 052732 162716 000002      MOVVB    #307,@(SP)     ;WRITE ERROR NUMBER IN CALL
226 052736 004736                SUB      #2,(SP)        ;MOVE SP TO RETURN IF ERROR
227 052740 162716 000010      JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
228 052744 000240                SUB      #10,(SP)       ;RESTORE SP TO NO ERROR
NOP
    
```



```

286 053232 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
287 053240 042737 040000 001140      BIC      #UNS,$GDDAT      ;
288 053246 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
289 053254 062716 000004      ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
290 053260 112776 000313 000000      MOVVB   #313,@(SP)        ;WRITE ERROR NUMBER
291 053266 162716 000002      SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
292 053272 004736      JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
293 053274 162716 000010      SUB      #10,(SP)         ;RESTORE SP TO NO ERROR
294 053300      190$:
295
296      ;REPORT ANY DRIVE TIMING ERROR, I.E., 'DTE' = 1
297 053300 032737 010000 001352      BIT      #DTE,RMER1I      ;IS DTE SET??
298 053306 001423      BEQ      200$            ;NO!!
299 053310 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
300 053316 042737 010000 001140      BIC      #DTE,$GDDAT      ;
301 053324 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
302 053332 062716 000004      ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
303 053336 112776 000314 000000      MOVVB   #314,@(SP)        ;WRITE ERROR NUMBER IN CALL
304 053344 162716 000002      SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
305 053350 004736      JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
306 053352 162716 000010      SUB      #10,(SP)         ;MOVE SP TO NO ERROR
307 053356      200$:
308
309      ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
310      ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
311 053356 032737 004000 001352      BIT      #WLE,RMER1I      ;WAS 'WLE' SET??
312 053364 001441      BEQ      220$            ;NO!!
313 053366 013737 001352 001142      MOV      RMER1I,$BDDAT      ;RECEIVED STATUS
314 053374 013737 001352 001140      MOV      RMER1I,$GDDAT      ;EXPECTED STATUS
315 053402 052737 004000 001140      BIS      #WLE,$GDDAT      ;
316 053410 062716 000004      ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
317 053414 112776 000315 000000      MOVVB   #315,@(SP)        ;WRITE ERROR NUMBER IN CALL
318 053422 032737 004000 001350      BIT      #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED??
319 053430 001404      BEQ      205$            ;NO!!
320 053432 032737 000010 001412      BIT      #BIT3,RMCS10     ;WAS COMMAND A WRITE??
321 053440 001406      BEQ      210$            ;YES!!
322 053442 112776 000316 000000 205$:  MOVVB   #316,@(SP)        ;CHANGE ERROR NUMBER
323 053450 042737 004000 001140      BIC      #WLE,$GDDAT      ;
324 053456 162716 000002 210$:  SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
325 053462 004736      JSR      PC,@(SP)+        ;REPORT ERROR AND RETURN
326 053464 162716 000010      SUB      #10,(SP)         ;MOVE SP TO NO ERROR
327
328 053470      220$:
329
330      ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
331 053470 062716 000004      ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR
332 053474 105776 000000      TSTB    @(SP)            ;WAS ERROR DETECTED??
333 053500 001404      BEQ      225$            ;NO - DO DATA CHECKS
334 053502 162716 000004      SUB      #4,(SP)          ;RESTORE SP
335 053506 000137 054510      JMP      340$            ;SKIP DATA CHECKS
336 053512 162716 000004 225$:  SUB      #4,(SP)          ;RESTORE SP
337
338      ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
339      ;IF HEADER COMPARE IS NOT INHIBITED
340 053516 013737 001412 055100      MOV      RMCS10,510$      ;STRIP AND STORE FUNCTION CODE
341 053524 042737 177700 055100      BIC      #^CFNCMSK,510$   ;
342 053532 022737 000063 055100      CMP      #WH.GO,510$      ;WAS FUNCTION WRITE HEADER & DATA??
    
```



```

343 053540 001512          BEQ      250$          ;YES - SKIP HEADER CHECKS
344 053542 032737 002000 001370 BIT      #HCI,RMOFI      ;WAS HCI SET??
345 053550 001106          BNE      250$          ;YES - SKIP HEADER CHECKS
346
347 ;SEE IF ANY HEADER ERRORS ARE SET, I.E., 'FER' OR 'HCRC' OR 'HCE'
348 053552 032737 000620 001352 BIT      #HCRC!FER!HCE,RMER1I
349 053560 001533          BEQ      270$          ;NO ERRORS SET
350
351 ;REPORT HEADER CRC ERROR IF SET
352 053562 032737 000400 001352 BIT      #HCRC,RMER1I      ;WAS HCRC SET??
353 053570 001422          BEQ      230$          ;NO!!
354 053572 013737 001352 001140 MOV      RMER1I,$GDDAT    ;EXPECTED STATUS
355 053600 042737 000400 001140 BIC      #HCRC,$GDDAT
356 053606 013737 001352 001142 MOV      RMER1I,$BDDAT    ;RECEIVED STATUS
357 053614 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR
358 053620 112776 000317 000000 MOVVB    #317,@(SP)       ;WRITE ERROR NUMBER
359 053626 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
360 053632 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
361 053634 000501          BR       260$
362 053636          230$:
363
364 ;REPORT FORMAT ERROR IF SET
365 053636 032737 000020 001352 BIT      #FER,RMER1I      ;WAS 'FER' SET??
366 053644 001422          BEQ      240$          ;NO!!
367 053646 013737 001352 001140 MOV      RMER1I,$GDDAT    ;EXPECTED STATUS
368 053654 042737 000020 001140 BIC      #FER,$GDDAT
369 053662 013737 001352 001142 MOV      RMER1I,$BDDAT    ;RECEIVED STATUS
370 053670 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USERS ERROR
371 053674 112776 000320 000000 MOVVB    #320,@(SP)       ;WRITE ERROR NUMBER
372 053702 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
373 053706 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
374 053710 000453          BR       260$
375 053712          240$:
376
377 ;REPORT HEADER COMPARE ERROR IF SET
378 053712 032737 000200 001352 BIT      #HCE,RMER1I      ;WAS 'HCE' SET??
379 053720 001453          BEQ      270$          ;NO!!
380 053722 013737 001352 001140 MOV      RMER1I,$GDDAT    ;EXPECTED STATUS
381 053730 042737 000200 001140 BIC      #HCE,$GDDAT
382 053736 013737 001352 001142 MOV      RMER1I,$BDDAT    ;RECEIVED STATUS
383 053744 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR
384 053750 112776 000321 000000 MOVVB    #321,@(SP)       ;WRITE ERROR NUMBER
385 053756 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
386 053762 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN
387 053764 000425          BR       260$
388 ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
389 ; .COMMAND WAS WRITE HEADER AND DATA, OR
390 ; .HEADER COMPARE INHIBIT WAS SET
391 053766 032737 000620 001352 250$: BIT      #HCE!FER!HCRC,RMER1I
392 053774 001425          BEQ      270$          ;NO ERRORS WERE SET
393 053776 013737 001352 001140 MOV      RMER1I,$GDDAT    ;EXPECTED STATUS
394 054004 042737 000620 001140 BIC      #HCE,FER!HCRC,$GDDAT
395 054012 013737 001352 001142 MOV      RMER1I,$BDDAT    ;RECEIVED STATUS
396 054020 062716 000004          ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
397 054024 112776 000322 000000 MOVVB    #322,@(SP)       ;WRITE ERROR NUMBER
398 054032 162716 000002          SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
399 054036 004736          JSR      PC,@(SP)+      ;REPORT ERROR AND RETURN

```

```

400 054040 162716 000010      260$: SUB    #10,(SP)      ;MOVE SP TO NO ERROR
401 054044 000137 054510      JMP    340$          ;OMIT FURTHER DATA CHECKS
402
403 054050      270$:
404
405      ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
406      ;DO READ ERROR CHECKS
407 054050 032737 000010 055100      BIT    #BIT3,510$    ;WAS THIS A WRITE COMMAND?
408 054056 001002      BNE    275$          ;NO..
409 054060 000137 054276      JMP    310$          ;GO DO WRITE STATUS CHECK
410 054064      275$:
411
412      ;REPORT DATA CHECK IF SET
413 054064 032737 100000 001352      BIT    #DCK,RMER1I  ;DATA CHECK ERROR??
414 054072 001450      BEQ    290$          ;NO!!
415 054074 013737 001352 001140      MOV    RMER1I,$GDDAT ;EXPECTED STATUS
416 054102 042737 100000 001140      BIC    #DCK,$GDDAT
417 054110 013737 001352 001142      MOV    RMER1I,$BDDAT ;RECEIVED STATUS
418 054116 062716 000004      ADD    #4,(SP)       ;MOVE SP TO USER'S ERROR
419 054122 112776 000323 000000      MOVB   #323,@(SP)    ;WRITE ERROR NUMBER
420 054130 032737 004000 001370      BIT    #ECI,RMOFI    ;WAS ECC CORRECTION DISABLED??
421 054136 001021      BNE    280$          ;YES!
422 054140 112776 000324 000000      MOVB   #324,@(SP)    ;CHANGE TO RECOVERABLE ERROR
423 054146 032737 000100 001352      BIT    #ECH,RMER1I  ;IS ERROR RECOVERABLE??
424 054154 001007      BNE    276$          ;NO !!
425      ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
426 054156 032737 000020 055100      BIT    #BIT4,510$    ;WAS THIS A READ COMMAND ??
427 054164 001406      BEQ    280$          ;NO !!
428 054166 162716 000004      SUB    #4,(SP)       ;RESTORE SP
429 054172 000410      BR     290$          ;SKIP ERROR - DATA WILL BE CORRECTED
430 054174 112776 000325 000000      276$: MOVB   #325,@(SP)    ;CHANGE TO NON RECOVERABLE
431 054202 162716 000002      280$: SUB    #2,(SP)       ;MOVE SP TO RETURN IF ERROR
432 054206 004736      JSR    PC,@(SP)+    ;REPORT ERROR AND RETURN
433 054210 162716 000010      SUB    #10,(SP)      ;RESTORE SP TO NO ERROR
434
435 054214      290$:
436
437      ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE - 1
438 054214 032737 000400 001346      BIT    #MDPE,RMCS2I ;PARITY ERROR SET??
439 054222 001423      BEQ    300$          ;NO!!
440 054224 013737 001346 001140      MOV    RMCS2I,$GDDAT ;EXPECTED STATUS
441 054232 042737 000400 001140      BIC    #MDPE,$GDDAT
442 054240 013737 001346 001142      MOV    RMCS2I,$BDDAT ;RECEIVED STATUS
443 054246 062716 000004      ADD    #4,(SP)       ;MOVE SP TO USER'S ERROR
444 054252 112776 000326 000000      MOVB   #326,@(SP)    ;WRITE ERROR NUMBER
445 054260 162716 000002      SUB    #2,(SP)       ;MOVE SP TO RETURN IF ERROR
446 054264 004736      JSR    PC,@(SP)+    ;REPORT ERROR AND RETURN
447 054266 162716 000010      SUB    #10,(SP)      ;MOVE SP TO NO ERROR
448 054272 000137 054510      300$: JMP    340$          ;SKIP WRITE STATUS CHECK
449
450 054276      310$:
451
452      ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF 'OM' 1
453 054276 032737 000001 001350      BIT    #OM,RMSI     ;IS OFFSET ON??
454 054304 001423      BEQ    320$          ;NO
455 054306 013737 001350 001140      MOV    RMSI,$GDDAT  ;EXPECTED STATUS
456 054314 042737 000001 001140      BIC    #OM,$GDDAT
    
```

```

457 054322 013737 001350 001142      MOV      RMDSI,$BDDAT      ;RECEIVED STATUS
458 054330 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
459 054334 112776 000327 000000      MOVVB   #327,@(SP)       ;WRITE ERROR NUMBER IN CALL
460 054342 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
461 054346 004736              JSR     PC,@(SP)+         ;REPORT ERROR AND RETURN
462 054350 162716 000010              SUB      #10,(SP)        ;MOVE SP TO NO ERROR
463 054354              320$:
464
465              ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF 'DPE' - 1
466 054354 032737 000010 001400      BIT      #DPE,RMER2I     ;DATA PARITY ERROR??
467 054362 001423              BEQ     330$             ;NO!!
468 054364 013737 001400 001140      MOV      RMER2I,$GDDAT   ;EXPECTED STATUS
469 054372 042737 000010 001140      BIC     #DPE,$GDDAT
470 054400 013737 001400 001142      MOV      RMER2I,$BDDAT   ;RECEIVED STATUS
471 054406 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
472 054412 112776 000330 000000      MOVVB   #330,@(SP)       ;WRITE ERROR NUMBER
473 054420 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
474 054424 004736              JSR     PC,@(SP)+         ;REPORT ERROR AND RETURN
475 054426 162716 000010              SUB      #10,(SP)        ;MOVE SP TO NO ERROR
476 054432              330$:
477
478              ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF 'WCF' 1
479 054432 032737 000040 001352      BIT      #WCF,RMER1I     ;IS 'WCF' SET??
480 054440 001423              BEQ     340$             ;NO!!
481 054442 013737 001352 001140      MOV      RMER1I,$GDDAT   ;EXPECTED STATUS
482 054450 042737 000040 001140      BIC     #WCF,$GDDAT
483 054456 013737 001352 001142      MOV      RMER1I,$BDDAT   ;RECEIVED STATUS
484 054464 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
485 054470 112776 000331 000000      MOVVB   #331,@(SP)       ;WRITE ERROR NUMBER
486 054476 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
487 054502 004736              JSR     PC,@(SP)+         ;REPORT ERROR AND RETURN
488 054504 162716 000010              SUB      #10,(SP)        ;MOVE SP TO NO ERROR
489 054510              340$:
490
491              ;REPORT 'DATA LATE' ERROR IF 'DLT' 1
492 054510 032737 100000 001346      BIT      #DLT,RMCS2I     ;IS 'DLT' SET??
493 054516 001423              BEQ     350$             ;NO!!
494 054520 013737 001346 001140      MOV      RMCS2I,$GDDAT   ;EXPECTED STATUS
495 054526 042737 100000 001140      BIC     #DLT,$GDDAT
496 054534 013737 001346 001142      MOV      RMCS2I,$BDDAT   ;RECEIVED STATUS
497 054542 062716 000004              ADD      #4,(SP)          ;MOVE SP TO USERS ERROR CALL
498 054546 112776 000332 000000      MOVVB   #332,@(SP)       ;WRITE ERROR NUMBER
499 054554 162716 000002              SUB      #2,(SP)          ;MOVE SP TO RETURN IF ERROR
500 054560 004736              JSR     PC,@(SP)+         ;REPORT ERROR AND RETURN
501 054562 162716 000010              SUB      #10,(SP)        ;MOVE SP TO NO ERROR
502 054566              350$:
503
504              ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
504 054566 013746 001350              MOV      RMDSI,-(SP)      ;STACK DRIVE STATUS
505 054572 042716 147677              BIC     #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
506 054576 022726 010100              CMP     #MOL!VV,(SP)+    ;IS DRIVE STATUS OK??
507 054602 001522              BEQ     380$             ;YES!.
508
509              ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
510              ;I.E. PIP - 1 AND SKI = 0
511 054604 032737 020000 001350      BIT      #PIP,RMDSI      ;IS 'PIP' SET??
512 054612 001430              BEQ     360$             ;NO!.
513 054614 032737 040000 001400      BIT      #SKI,RMER2I     ;WAS 'SKI' ERROR REPORTED??

```

```

514 054622 001024      BNE      360$      ;YES-DONT REPORT PIP
515 054624 013737 001350 001140      MOV      RMDSI,$GDDAT ;EXPECTED STATUS
516 054632 042737 020000 001140      BIC      #PIP,$GDDAT
517 054640 013737 001350 001142      MOV      RMDSI,$BDDAT ;RECEIVED STATUS
518 054646 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
519 054652 112776 000333 000000      MOV      #333,@(SP)  ;WRITE ERROR NUMBER
520 054660 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
521 054664 004736      JSR      PC,@(SP)+   ;REPORT ERROR AND RETURN
522 054666 162716 000010      SUB      #10,(SP)    ;MOVE SP TO NO ERROR
523 054672 000240
524 054674      360$:
525
526      ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
527      ;REPORTED, I.E., MOL = OPI = 0
528 054674 032737 010000 001350      BIT      #MOL,RMDSI  ;IS MEDIUM ON LINE??
529 054702 001027      BNE      370$      ;YES!!
530 054704 032737 020000 001352      BIT      #OPI,RMER1I ;WAS OPI ERROR REPORTED??
531 054712 001023      BNE      370$      ;YES!!
532 054714 013737 001350 001140      MOV      RMDSI,$GDDAT ;EXPECTED STATUS
533 054722 052737 010000 001140      BIS      #MOL,$GDDAT
534 054730 013737 001350 001142      MOV      RMDSI,$BDDAT ;RECEIVED STATUS
535 054736 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USER'S ERROR
536 054742 112776 000334 000000      MOV      #334,@(SP)  ;WRITE ERROR NUMBER
537 054750 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
538 054754 004736      JSR      PC,@(SP)+   ;REPORT ERROR AND RETURN
539 054756 162716 000010      SUB      #10,(SP)    ;MOVE SP TO NO ERROR
540 054762      370$:
541
542      ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
543      ;REPORTED, I.E., VV = IVC = 0
544 054762 032737 000100 001350      BIT      #VV,RMDSI   ;IS VOLUME VALID??
545 054770 001027      BNE      380$      ;YES!!
546 054772 032737 010000 001400      BIT      #IVC,RMER2I ;WAS IVC ERROR REPORTED??
547 055000 001033      BNE      390$      ;YES!!
548 055002 013737 001350 001140      MOV      RMDSI,$GDDAT ;EXPECTED STATUS
549 055010 052737 000100 001140      BIS      #VV,$GDDAT
550 055016 013737 001350 001142      MOV      RMDSI,$BDDAT ;RECEIVED STATUS
551 055024 062716 000004      ADD      #4,(SP)      ;MOVE SP TO USERS ERROR CALL
552 055030 112776 000335 000000      MOV      #335,@(SP)  ;WRITE ERROR NUMBER
553 055036 162716 000002      SUB      #2,(SP)      ;MOVE SP TO RETURN IF ERROR
554 055042 004736      JSR      PC,@(SP)+   ;REPORT ERROR AND RETURN
555 055044 162716 000010      SUB      #10,(SP)    ;MOVE SP TO NO ERROR
556 055050      380$:
557
558      ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
559 055050 062716 000004      ADD      #4,(SP)      ;MOVE SP TO ERROR CALL
560 055054 105776 000000      TSTB    @(SP)        ;ANY ERROR??
561 055060 001403      BEQ      390$      ;NO. !
562 055062 062716 000004      ADD      #4,(SP)      ;YES - MOVE SP TO ERROR RETURN
563 055066 000402      BR       400$
564 055070 162716 000004      390$: SUB      #4,(SP)    ;MOVE SP TO NO ERROR RETURN
565
566 055074 000207      400$: RTS      PC      ;RETURN TO USER
567
568 055076 000000      500$: .WORD      ;ERROR FLAGS
569 055100 000000      510$: .WORD      ;TEMPORARY STORAGE

```

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

055102
055102 062716 000004
055106 105076 000000
055112 162716 000004
055116 013746 001350
055122 042716 147677
055126 022726 010100
055132 001524
055134 032737 010000 001350
055142 001030
055144 032737 020000 001352
055152 001024
055154 013737 001350 001140
055162 052737 010000 001140
055170 013737 001350 001142
055176 062716 000004
055202 112776 000207 000000
055210 162716 000002
055214 004736
055216 162716 000010
055222 000240
055224
055224 032737 000100 001350
055232 001030

```
.SBITL STATIC DRIVE STATUS CHECK SUBROUTINE
:THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
:STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID. THE SUBROUTINE
:CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
:SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
:THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
:IF TRUE:
:
:   .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
:   THAT MOL IS ASSUMED TO HAVE BEEN SET
:   .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
:   .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
:   .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
:   .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
:THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
:(1) JSR PC,STCDRVSTS
:     BR      ???          RETURN HERE IF NO ERROR
:     NOP          RETURN HERE TO REPORT AN ERROR
:     ERROR      ERROR NUMBER DEFINED BY SUB
:     JSR PC,@(SP)+    GO BACK TO SUB FOR MORE ERROR CHECKS
:     ???          RETURN HERE IF NO MORE ERRORS
STCDRVSTS:
:CLEAR USER'S ERROR CALL
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
CLRB @ (SP) ;CLEAR ERROR NUMBER
SUB #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
:SEF IF 'MOL' = 'VV' = 1, AND 'PIP' = 0
MOV RMDSI,-(SP) ;PUT DRIVE STATUS ON STACK
BIC #^C<PIP!MOL!VV>,(SP)
CMP #MOL!VV,(SP)+ ;ARE MOL,VV AND PIP O.K.??
BEQ 30$ ;YES!!
:REPORT AN ERROR IF MOL = 0 AND 'OPI' = 0
BIT #MOL,RMDSI ;IS MOL ON ??
BNE 10$ ;YES!!
BIT #OPI,RMER1I ;WAS 'OPI' SET??
BNE 10$ ;YES-DONT REPORT 'MOL' 0
MOV RMDSI,$GDDAT ;EXPECTED STATUS
BIS #MOL,$GDDAT
MOV RMDSI,$BDDAT ;REVEALED STATUS
ADD #4,(SP) ;MOVE SP TO USER'S ERROR CALL
MOVB #207,@(SP) ;WRITE ERROR NUMBER IN CALL
SUB #2,(SP) ;MOVE SP TO RETURN FOR ERROR
JSR PC,@(SP)+ ;REPORT ERROR VIA USER
SUB #10,(SP) ;MOVE SP BACK TO NO ERROR RETURN
NOP
10$:
:REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND 'IVC' = 0
BIT #VV,RMDSI ;IS 'VV' = 0??
BNE 20$ ;NO.!
```

```

58 055234 032737 010000 001400      BIT      #IVC,RMER2I      ;WAS 'IVC' SET??
59 055242 001024                    BNE      20$            ;YES-DONT REPORT 'VV'  0
60 055244 013737 001350 001140      MOV      RMDSI,$GDDAT   ;EXPECTED STATUS
61 055252 052737 000100 001350      BIS      #VV,RMDSI
62 055260 013737 001350 001142      MOV      RMDSI,$BDDAT   ;RECEIVED STATUS
63 055266 062716 000004                    ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
64 055272 112776 000210 000000      MOVVB   #210,@(SP)     ;WRITE ERROR NUMBER IN CALL
65 055300 162716 000002                    SUB      #2,(SP)        ;MOVE SP TO RETURN FOR ERROR
66 055304 004736                    JSR      PC,@(SP)+      ;REPORT ERROR VIA USER
67 055306 162716 000010                    SUB      #10,(SP)       ;MOVE SP BACK TO NO ERROR
68 055312 000240                    NOP
69 055314                    20$:
70
71                                ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND 'SKI' - 0
72 055314 032737 020000 001350      BIT      #PIP,RMDSI     ;IS DRIVE OFF CYLINDER??
73 055322 001430                    BEQ      30$            ;NO!!
74 055324 032737 040000 001400      BIT      #SKI,RMER2I    ;WAS 'SKI' SET??
75 055332 001024                    BNE      30$            ;YES-DONT REPORT 'PIP' - 1
76 055334 013737 001350 001140      MOV      RMDSI,$GDDAT   ;EXPECTED STATUS
77 055342 042737 020000 001140      BIC      #PIP,$GDDAT
78 055350 013737 001350 001142      MOV      RMDSI,$BDDAT   ;RECEIVED STATUS
79 055356 062716 000004                    ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
80 055362 112776 000211 000000      MOVVB   #211,@(SP)     ;WRITE ERROR NUMBER IN USER'S CALL
81 055370 162716 000002                    SUB      #2,(SP)        ;MOVE SP TO RETURN FOR ERROR
82 055374 004736                    JSR      PC,@(SP)+      ;REPORT ERROR VIA USER
83 055376 162716 000010                    SUB      #10,(SP)       ;MOVE SP TO NO ERROR RETURN
84 055402 000240                    NOP
85 055404                    30$:
86
87                                ;SEE IF 'SKI' - 'DVC' = 0
88 055404 013746 001400                    MOV      RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
89 055410 042726 137577                    BIC      #^C<SKI!DVC>,(SP)+
90 055414 001460                    BEQ      60$            ;BRANCH IF NO ERROR
91 055416                    40$:
92
93                                ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
94 055416 032737 000200 001400      BIT      #DVC,RMER2I    ;ANY DEVICE FAULT??
95 055424 001424                    BEQ      50$            ;NO!!
96 055426 013737 001400 001140      MOV      RMER2I,$GDDAT   ;EXPECTED STATUS
97 055434 042737 000200 001140      BIC      #DVC,$GDDAT
98 055442 013737 001400 001142      MOV      RMER2I,$BDDAT   ;RECEIVED STATUS
99 055450 062716 000004                    ADD      #4,(SP)        ;MOVE SP TO USER'S CALL
100 055454 112776 000212 000000      MOVVB   #212,@(SP)     ;WRITE NUMBER OF ERROR IN CALL
101 055462 162716 000002                    SUB      #2,(SP)        ;MOVE SP TO RETURN FOR ERROR
102 055466 004736                    JSR      PC,@(SP)+      ;REPORT ERROR VIA USER
103 055470 162716 000010                    SUB      #10,(SP)       ;MOVE SP BACK TO NO ERROR
104 055474 000240                    NOP
105 055476                    50$:
106
107                                ;REPORT AN ERROR IF 'SKI' = 1
108 055476 032737 040000 001400      BIT      #SKI,RMER2I    ;IS THERE A SEEK INCOMPLETE ERROR
109 055504 001424                    BEQ      60$            ;NO!!
110 055506 013737 001400 001140      MOV      RMER2I,$GDDAT   ;EXPECTED STATUS
111 055514 042737 040000 001140      BIC      #SKI,$GDDAT
112 055522 013737 001400 001142      MOV      RMER2I,$BDDAT   ;RECEIVED STATUS
113 055530 062716 000004                    ADD      #4,(SP)        ;MOVE SP TO USER'S ERROR CALL
114 055534 112776 000213 000000      MOVVB   #213,@(SP)     ;WRITE ERROR NUMBER IN USER'S ERROR CALL
    
```

115	055542	162716	000002		SUB	#2,(SP)		;MOVE SP TO RETURN FOR ERROR
116	055546	004736			JSR	PC,@(SP)+		;REPORT ERROR VIA USER
117	055550	162716	000010		SUB	#10,(SP)		;MOVE SP BACK TO NO ERROR
118	055554	000240			NOP			
119	055556			60\$:				
120								
121								
122	055556	062716	000004					
123	055562	105776	000000		ADD	#4,(SP)		;MOVE SP TO USER'S ERROR CALL
124	055566	0C1403			TSTB	@(SP)		;WAS AN ERROR DETECTED??
125	055570	062716	000004		BEQ	70\$;NO!!
126	055574	000402			ADD	#4,(SP)		;YES - MOVE SP TO USER'S ERROR RETURN
127	055576	162716	000004		BR	80\$		
128	055602	000240		70\$:	SUB	#4,(SP)		;NO - MOVE SP TO NO ERROR RETURN
129	055604	000207		80\$:	NOP			
					R'S	PC		;RETURN TO USER

.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

```

```

055606          $SAVREG:
055606 010046    MOV     R0,-(SP)      ;;PUSH R0 ON STACK
055610 010146    MOV     R1,-(SP)      ;;PUSH R1 ON STACK
055612 010246    MOV     R2,-(SP)      ;;PUSH R2 ON STACK
055614 010346    MOV     R3,-(SP)      ;;PUSH R3 ON STACK
055616 010446    MOV     R4,-(SP)      ;;PUSH R4 ON STACK
055620 010546    MOV     R5,-(SP)      ;;PUSH R5 ON STACK
055622 016646 000022    MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
055626 016646 000022    MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
055632 016646 000022    MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
055636 016646 000022    MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
055642 000002    RTI

```

*RESTORE R0-R5

*CALL:
* RESREG

```

055644          $RESREG:
055644 012666 000022    MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
055650 012666 000022    MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
055654 012666 000022    MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
055660 012666 000022    MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
055664 012605    MOV     (SP)+,R5        ;;POP STACK INTO R5
055666 012604    MOV     (SP)+,R4        ;;POP STACK INTO R4
055670 012603    MOV     (SP)+,R3        ;;POP STACK INTO R3
055672 012602    MOV     (SP)+,R2        ;;POP STACK INTO R2
055674 012601    MOV     (SP)+,R1        ;;POP STACK INTO R1
055676 012600    MOV     (SP)+,R0        ;;POP STACK INTO R0
055700 000002    RTI

```


.SBTTL BINARY TO ASCII AND TYPE ROUTINE

 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
 *BINARY-ASCII NUMBER AND TYPE IT.

*CALL:

* MOV NUMBER,-(SP) ::NUMBER TO BE TYPED
 * TYPBN ::TYPE IT

055702	010146			\$TYPBN:	MOV	R1,-(SP)	::SAVE R1 ON THE STACK
055704	016601	000006			MOV	6(SP),R1	::GET THE INPUT NUMBER
055710	000261				SEC		::SET 'C' SO CAN KEEP TRACK OF THE NUMBER OF BITS
055712	112737	000060	055754	1\$:	MOVB	#'0,\$BIN	::SET CHARACTER TO AN ASCII '0'.
055720	006101				ROL	R1	::GET THIS BIT
055722	001406				BEQ	2\$::DONE?
055724	105537	055754			ADCB	\$BIN	::NO--SET THE CHARACTER EQUAL TO THIS BIT
055730	104401	055754			TYPE	,\$BIN	::GO TYPE THIS BIT
055734	000241				CLC		::CLEAR 'C' SO CAN KEEP TRACK OF BITS
055736	000765				BR	1\$::GO DO THE NEXT BIT
055740	012601			2\$:	MOV	(SP)+,R1	::POP THE STACK INTO R1
055742	016666	000002	000004		MOV	2(SP),4(SP)	::ADJUST THE STACK
055750	012616				MOV	(SP)+,(SP)	
055752	000002				RTI		::RETURN TO USER
055754	000	000		\$BIN:	.BYTE	0,0	::STORAGE FOR ASCII CHAR. AND TERMINATOR

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

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

```

055756          $TYPDS·
055756 010046   MOV      R0,-(SP)      ;;PUSH R0 ON STACK
055760 010146   MOV      R1,-(SP)      ;;PUSH R1 ON STACK
055762 010246   MOV      R2,-(SP)      ;;PUSH R2 ON STACK
055764 010346   MOV      R3,-(SP)      ;;PUSH R3 ON STACK
055766 010546   MOV      R5,-(SP)      ;;PUSH R5 ON STACK
055770 012746 020200   MOV      #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
055774 016605 000020   MOV      20(SP),R5        ;;GET THE INPUT NUMBER
056000 100004   BPL      1$                ;;BR IF INPUT IS POS.
056002 005405   NEG      R5                ;;MAKE THE BINARY NUMBER POS.
056004 112766 000055 000001   MOVB    #'-,1(SP)        ;;MAKE THE ASCII NUMBER NEG.
056012 005000   CLR      R0                ;;ZERO THE CONSTANTS INDEX
056014 012703 056172   MOV      #SDBLK,R3        ;;SETUP THE OUTPUT POINTER
056020 112723 000040   MOVB    #' ,(R3)+        ;;SET THE FIRST CHARACTER TO A BLANK
056024 005002   CLR      R2                ;;CLEAR THE BCD NUMBER
056026 016001 056162   MOV      $DTBL(R0),R1     ;;GET THE CONSTANT
056032 160105   SUB      R1,R5            ;;FORM THIS BCD DIGIT
056034 002402   BLT     4$                ;;BR IF DONE
056036 005202   INC     R2                ;;INCREASE THE BCD DIGIT BY 1
056040 000774   BR      3$
056042 060105   4$:   ADD     R1,R5            ;;ADD BACK THE CONSTANT
056044 005702   TST     R2                ;;CHECK IF BCD DIGIT=0
056046 001002   BNE     5$                ;;FALL THROUGH IF 0
056050 105716   TSTB   (SP)                ;;STILL DOING LEADING 0'S?
056052 100407   BMI     7$                ;;BR IF YES
056054 106316   5$:   ASLB   (SP)                ;;MSD?
056056 103003   BCC     6$                ;;BR IF NO
056060 116663 000001 177777   MOVB    1(SP),-1(R3)      ;;YES--SET THE SIGN
056066 052702 000060   BIS     #'0,R2            ;;MAKE THE BCD DIGIT ASCII
056072 052702 000040   7$:   BIS     #' ,R2            ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
056076 110223   MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
056100 005720   TST     (R0)+            ;;JUST INCREMENTING
056102 020027 000010   CMP     R0,#10           ;;CHECK THE TABLE INDEX
056106 002746   BLT     2$                ;;GO DO THE NEXT DIGIT
056110 003002   BGT     8$                ;;GO TO EXIT
056112 010502   MOV     R5,R2            ;;GET THE LSD
056114 000764   BR      6$                ;;GO CHANGE TO ASCII
056116 105726   8$:   TSTB   (SP)+            ;;WAS THE LSD THE FIRST NON-ZERO?
056120 100003   BPL     9$                ;;BR IF NO
056122 116663 177777 177776   MOVB    -1(SP),-2(R3)    ;;YES--SET THE SIGN FOR TYPING
056130 105013   9$:   CLRB   (R3)                ;;SET THE TERMINATOR
056132 012605   MOV     (SP)+,R5        ;;POP STACK INTO R5
056134 012603   MOV     (SP)+,R3        ;;POP STACK INTO R3
056136 012602   MOV     (SP)+,R2        ;;POP STACK INTO R2
056140 012601   MOV     (SP)+,R1        ;;POP STACK INTO R1
    
```

```
056142 012600      MOV      (SP)+,R0      ;;POP STACK INTO RC
056144 104401 056172 TYPE      $DBLK      ;;NOW TYPE THE NUMBER
056150 016666 000002 000004 MOV      2(SP),4(SP)  ;;ADJUST THE STACK
056156 012616      MOV      (SP)+,(SP)
056160 000002      RTI                          ;;RETURN TO USER
056162 023420      $DTBL: 10000.
056164 001750      1000.
056166 000144      100.
056170 000012      10.
056172      $DBLK: .BLKW 4
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

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

```

```

*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC

```

```

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

```

```

*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

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

```

056202	017646	000000		\$TYPOS:	MOV	@(SP),-(SP)	;;PICKUP THE MODE
056206	116637	000001	056425		MOVB	1(SP), \$OFILL	;;LOAD ZERO FILL SWITCH
056214	112637	056427			MOVB	(SP)+, \$OMODE+1	;;NUMBER OF DIGITS TO TYPE
056220	062716	000002			ADD	#2,(SP)	;;ADJUST RETURN ADDRESS
056224	000406				BR	\$TYPON	
056226	112737	000001	056425	\$TYPOC:	MOVB	#1,\$OFILL	;;SET THE ZERO FILL SWITCH
056234	112737	000006	056427		MOVB	#6,\$OMODE+1	;;SET FOR SIX(6) DIGITS
056242	112737	000005	056424	\$TYPON:	MOVB	#5,\$OCNT	;;SET THE ITERATION COUNT
056250	010346				MOV	R3,-(SP)	;;SAVE R3
056252	010446				MOV	R4,-(SP)	;;SAVE R4
056254	010546				MOV	R5,-(SP)	;;SAVE R5
056256	113704	056427			MOVB	\$OMODE+1,R4	;;GET THE NUMBER OF DIGITS TO TYPE
056262	005404				NEG	R4	
056264	062704	000006			ADD	#6,R4	;;SUBTRACT IT FOR MAX. ALLOWED
056270	110437	056426			MOVB	R4,\$OMODE	;;SAVE IT FOR USE
056274	113704	056425			MOVB	\$OFILL,R4	;;GET THE ZERO FILL SWITCH
056300	016605	000012			MOV	12(SP),R5	;;PICKUP THE INPUT NUMBER
056304	005003				CLR	R3	;;CLEAR THE OUTPUT WORD
056310	000404			1\$:	ROL	R5	;;ROTATE MSB INTO 'C'
056312	006105				BR	3\$;;GO DO MSB
056314	006105			2\$:	ROL	R5	;;FORM THIS DIGIT
056316	006105				ROL	R5	
056320	010503				MOV	R5,R3	
056322	006103			3\$:	ROL	R3	;;GET LSB OF THIS DIGIT
056324	105337	056426			DECB	\$OMODE	;;TYPE THIS DIGIT?
056330	100016				BPL	7\$;;BR IF NO
056332	042703	177770			BIC	#177770,R3	;;GET RID OF JUNK
056336	001002				BNE	4\$;;TEST FOR 0
056340	005704				TST	R4	;;SUPPRESS THIS 0?
056342	001403				BEQ	5\$;;BR IF YES
056344	005204			4\$:	INC	R4	;;DON'T SUPPRESS ANYMORE 0'S

056346	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
056352	052703	000040	5\$:	BIS	#' ,R3	::MAKE ASCII IF NOT ALREADY
056356	110337	056422		MOVB	R3,8\$::SAVE FOR TYPING
056362	104401	056422		TYPE	.8\$::GO TYPE THIS DIGIT
056366	105337	056424	7\$:	DECB	\$OCNT	::COUNT BY 1
056372	003347			BGT	2\$::BR IF MORE TO DO
056374	002402			BLT	6\$::BR IF DONE
056376	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
056400	000744			BR	2\$::GO DO THE LAST DIGIT
056402	012605		6\$:	MOV	(S)+,R5	::RESTORE R5
056404	012604			MOV	(SP)+,R4	::RESTORE R4
056406	012603			MOV	(SP)+,R3	::RESTORE R3
056410	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
056416	012616			MOV	(SP)+,(SP)	
056420	000002			RTI		::RETURN
056422	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
056423	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
056424	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
056425	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
056426	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

.SBTTL TYPE ROUTINE

```

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

```

```

056430 105737 001173 $TYPE: TSTB $TPFLG      ;;IS THERE A TERMINAL?
056434 100002      BPL      1$      ;;BR IF YES
056436 000000      HALT      ;;HALT HERE IF NO TERMINAL
056440 000430      BR      3$      ;;LEAVE
056442 010046      1$: MOV      RO,-(SP)  ;;SAVE RO
056444 017600 000002 MOV      @2(SP),RO  ;;GET ADDRESS OF ASCIZ STRING
056450 122737 000001 001242 CMPB     #APTENV,$ENV  ;;RUNNING IN APT MODE
056456 001011      BNE      62$     ;;NO,GO CHECK FOR APT CONSOLE
056460 132737 000100 001243 BITB     #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
056466 001405      BEQ      62$     ;;NO,GO CHECK FOR CONSOLE
056470 010037 056500 MOV      RO,61$     ;;SETUP MESSAGE ADDRESS FOR APT
056474 004737 063106 JSR      PC,$ATY3   ;;SPOOL MESSAGE TO APT
056500 000000      61$: .WORD    0      ;;MESSAGE ADDRESS
056502 132737 000040 001243 62$: BITB     #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
056510 001003      BNE      60$     ;;YES,SKIP TYPE OUT
056512 112046      2$: MOVB     (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPE1 ONTO STACK
056514 001005      BNE      4$      ;;BR IF IT ISN'T THE TERMINATOR
056516 005726      TST      (SP)+   ;;IF TERMINATOR POP IT OFF THE STACK
056520 012600      60$: MOV      (SP)+,RO  ;;RESTORE RO
056522 062716 000002 3$: ADD      #2,(SP)  ;;ADJUST RETURN PC
056526 000002      RTI      ;;RETURN
056530 122716 000011 4$: CMPB     #HT,(SP)  ;;BRANCH IF <HT>
056534 001430      BEQ      8$      ;;BRANCH IF NOT <CRLF>
056536 122716 000200 CMPB     #CRLF,(SP)
056542 001006      BNE      5$      ;;POP <CR><LF> EQUIV
056544 005726      TST      (SP)+   ;;TYPE A CR AND LF
056546 104401      TYPE     $CRLF
056550 001217      CLRB     $CHARCNT  ;;CLEAR CHARACTER COUNT
056552 105037 056760 BR      2$      ;;GET NEXT CHARACTER
056556 000755      JSR      PC,$TYPEC  ;;GO TYPE THIS CHARACTER
056560 004737 056642 5$: CMPB     $FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
056564 123726 001172 6$: BNE      2$      ;;IF NO GO GET NEXT CHAR.
056570 001350      MOV      $NULL,-(SP)  ;;GET # OF FILLER CHARS. NEEDED
056572 013746 001170      AND     THE NULL CHAR.
056576 105366 000001 7$: DECB     1(SP)  ;;DOES A NULL NEED TO BE TYPED?
056602 002770      BLT      6$      ;;BR IF NO--GO POP THE NULL OFF OF STACK
056604 004737 056642 JSR      PC,$TYPEC  ;;GO TYPE A NULL
056610 105337 056760 DECB     $CHARCNT  ;;DO NOT COUNT AS A COUNT
056614 000770      BR      7$      ;;LOOP

```

;HORIZONTAL TAB PROCESSOR

```

056616 112716 000040      8$:  MOVB  #' (SP)      ;; REPLACE TAB WITH SPACE
056622 004737 056642      9$:  JSR  PC,$TYPEC      ;; TYPE A SPACE
056626 132737 000007 056760  BITB  #7,$CHARCNT      ;; BRANCH IF NOT AT
056634 001372           BNE  9$                ;; TAB STOP
056636 005726           TST  (SP)+              ;; POP SPACE OFF STACK
056640 000724           BR   2$                ;; GET NEXT CHARACTER
056642                                     $TYPEC:
056642 105777 122312      TSTB  @STKS              ;; CHAR IN KYBD BUFFER?
056646 100022           BPL  10$                ;; BR IF NOT
056650 017746 122306      MOV   @STKB,-(SP)        ;; GET CHAR
056654 042716 177600      BIC   #177600,(SP)      ;; STRIP EXTRANEIOUS BITS
056660 122716 000023      CMPB  #$XOFF,(SP)      ;; WAS CHAR XOFF
056664 001012           BNE  102$             ;; BR IF NOT
056666                                     101$:
056666 105777 122266      TSTB  @STKS              ;; WAIT FOR CHAR
056672 100375           BPL  101$             ;;
056674 117716 122262      MOVB  @STKB,(SP)        ;; GET CHAR
056700 042716 177600      BIC   #177600,(SP)      ;; STRIP IT
056704 122716 000021      CMPB  #$XON,(SP)      ;; WAS IT XON?
056710 001366           BNE  101$             ;; BR IF NOT
056712                                     102$:
056712 005726           TST  (SP)+              ;; FIX STACK
056714                                     10$:
056714 105777 122244      TSTB  @STPS              ;; WAIT UNTIL PRINTER IS READY
056720 100375           BPL  10$                ;;
056722 116677 000002 122236  MOVB  2(SP),@STPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
056730 122766 000015 000002  CMPB  #CR,2(SP)        ;; IS CHARACTER A CARRIAGE RETURN?
056736 001003           BNE  1$                ;; BRANCH IF NO
056740 105037 056760      CLRB  $CHARCNT        ;; YES--CLEAR CHARACTER COUNT
056744 000406           BR   $TYPEX          ;; EXIT
056746 122766 000012 000002 1$:  CMPB  #LF,2(SP)        ;; IS CHARACTER A LINE FEED?
056754 001402           BEQ  $TYPEX          ;; BRANCH IF YES
056756 105227           INCB (PC)+          ;; COUNT THE CHARACTER
056760 000000      $CHARCNT: .WORD 0      ;; CHARACTER COUNT STORAGE
056762 000207      $TYPEX: RTS PC

```

.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
*SW08=1      LOOP ON TEST IN SWR<7:0>
*CALL
*          SCOPE          ;;SCOPE-IOT
    
```

```

056764          $SCOPE:
05C          104410          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
056766          004737          057532          JSR          PC,STOP
056772          032777          040000          122154          1$:          BIT          #BIT14,@SWR          ;;LOOP ON PRESENT TEST?
057000          001402          BEQ          9$          ;;NO IF SW14=0
057002          000137          057432          JMP          $OVER          ;;JUMP OVER SCOPE ROUTINE
057006          000416          9$:          ;;#####START OF CODE FOR THE XOR TESTER#####
          $XTSTR: BR          6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
          ;;THIS INSTRUCTION TO A 'NOP' (NOP 240)
057010          013746          000004          MOV          @ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
057014          012737          057034          000004          MOV          #5$,@ERRVEC          ;;SET FOR TIMEOUT
057022          005737          177060          TST          @177060          ;;TIME OUT ON XOR?
057026          012637          000004          MOV          (SP)+,@ERRVEC          ;;RESTORE THE ERROR VECTOR
057032          000561          BR          $SVLAD          ;;GO TO THE NEXT TEST
057034          022626          5$:          CMP          (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
057036          012637          000004          MOV          (SP)+,@ERRVEC          ;;RESTORE THE ERROR VECTOR
057042          000521          BR          7$          ;;LOOP ON THE PRESENT TEST
057044          6$:          ;;#####END OF CODE FOR THE XOR TESTER#####
057044          032777          000400          122102          BIT          #BIT08,@SWR          ;;LOOP ON SPEC. TEST?
057052          001421          BEQ          2$          ;;BR IF NO
057054          005046          CLR          -(SP)          ;;CLEAR A TEMP. LOCATION
057056          117716          122072          MOVB        @SWR,(SP)          ;;PICKUP THE DESIRED TEST NUMBER
057062          001414          BEQ          8$          ;;BRANCH IF BAD TEST NUMBER IN SWR
057064          022716          000031          CMP          #31,(SP)          ;;CHECK THE NUMBER IN THE SWR
057070          002411          BLT          8$          ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
057072          011637          001116          MOV          (SP),$TSTNM          ;;UPDATE THE TEST NUMBER
057076          005316          DEC          (SP)          ;;BACKUP BY ONE
057100          006316          ASL          (SP)          ;;SCALE THE TEST NUMBER AS AN INDEX
057102          062716          057450          ADD          #SW08TBL,(SP)          ;;FORM THE ADDRESS OF TEST POINTER
057106          013637          001122          MOV          @(SP)+,$LPADR          ;;SET LOOP ADDRESS TO DESIRED TEST
057112          000547          BR          $OVER          ;;GO LOOP ON THE TEST
057114          005726          8$:          TST          (SP)+          ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
057116          105737          001117          2$:          TSTB        $ERFLG          ;;HAS AN ERROR OCCURRED?
057122          001502          BEQ          3$          ;;BR IF NO
057124          022737          177777          060150          CMP          #-1,CPSAVE          ;;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED
057132          001455          BEQ          2003$          ;;KICK AROUND ROUTINE IF SO
057134          013746          000004          MOV          ERRVEC,-(SP)          ;;SAVE CONTENTS OF ERROR VECTOR
057140          012737          057156          000004          MOV          #2000$,ERRVEC          ;;SETUP 'TRAP' RETURN ADDRESS
057146          013737          177766          060150          MOV          177766,CPSAVE          ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
057154          000406          BR          2001$
057156          012737          177777          060150          2000$:          MOV          #-1,CPSAVE          ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
057164          012716          057172          MOV          #2001$,(SP)          ;;SETUP RETURN ADDRESS
    
```



```

SCOPE HANDLER ROUTINE

057170 000002          RTI
057172 012637 000004    2001$: MOV      (SP)+,ERRVEC      ;;RESTORE CONTENTS OF ERROR VECTOR

057176 022737 177777 060150 2002$: CMP      #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
057204 001430          BEQ      2003$      ;;BRANCH IF SO
057206 032737 000001 060150    BIT      #BIT00,CPSAVE      ;;SEE IF THE POWER MONITOR BIT IS ON
057214 001424          BEQ      2003$      ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
057216 042737 000001 177766    BIC      #BIT00,177766      ;;CLEAR THE BIT FOUND TO BE SET
057224 013746 001154          MOV      SWR,-(SP)      ;;SAVE SWR ADDRESS
057230 017646 000000          MOV      @ (SP),-(SP)      ;;SAVE SWR VALUE
057234 012737 000176 001154    MOV      #176,SWR      ;;GET SOFTWARE SWR ADDRESS
057242 011677 121706          MOV      (SP),@SWR      ;;GET CURRENT SWR VALUE
057246 042777 001000 121700    BIC      #BIT09,@SWR      ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
057254 104177          EMT      177      ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
057256 012676 000000          MOV      (SP)+,@(SP)      ;;RESTORE SWR TO ORIGINAL VALUE
057262 012637 001154          MOV      (SP)+,SWR      ;;RESTORE SWR ADDRESS
057266          2003$.
057266 123737 001131 001117    CMPB    $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
057274 101015          BHI     3$      ;;BR IF NO
057276 032777 001000 121650    BIT      #BIT09,@SWR      ;;LOOP ON ERROR?
057304 001404          BEQ     4$      ;;BR IF NO
057306 013737 001124 001122 7$:  MOV      $LPERR,$LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
057314 000446          BR      $OVER
057316 105037 001117          4$:  CLRB    $ERFLG      ;;ZERO THE ERROR FLAG
057322 005037 001206          CLR     $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
057326 000415          BR      1$      ;;ESCAPE TO THE NEXT TEST
057330 032777 004000 121616 3$:  BIT      #BIT11,@SWR      ;;INHIBIT ITERATIONS?
057336 001011          BNE    1$      ;;BR IF YES
057340 005737 001230          TST    $PASS      ;;IF FIRST PASS OF PROGRAM
057344 001406          BEQ    1$      ;;INHIBIT ITERATIONS
057346 005237 001120          INC    $ICNT      ;;INCREMENT ITERATION COUNT
057352 023737 001206 001120    CMP    $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
057360 002024          BGE    $OVER      ;;BR IF MORE ITERATION REQUIRED
057362 012737 000001 001120 1$:  MOV      #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
057370 013737 057446 001206    MOV    $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
057376 105237 001116          $SVLAD: INCB   $TSTNM      ;;COUNT TEST NUMBERS
057402 113737 001116 001226    MOVB   $TSTNM,$TESTN      ;;SET TEST NUMBER IN APT MAILBOX
057410 011637 001122          MOV    (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
057414 011637 001124          MOV    (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
057420 005037 001210          CLR    $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
057424 112737 000001 001131    MOVB   #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
057432 013777 001116 121516 $OVER: MOV    $TSTNM,@DISPLAY      ;;DISPLAY TEST NUMBER
057440 013716 001122          MOV    $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
057444 000002          RTI
057446 000005          $MXCNT: 5.      ;;FIXES PS
057450          $SWOBTBL:      ;;MAX. NUMBER OF ITERATIONS
          .REPT $TN-1
057450 010070          .WORD  TST1+2      ;;STARTING ADDRESS OF TEST 1
057452 010270          .WORD  TST2+2      ;;STARTING ADDRESS OF TEST 2
057454 011132          .WORD  TST3+2      ;;STARTING ADDRESS OF TEST 3
057456 011766          .WORD  TST4+2      ;;STARTING ADDRESS OF TEST 4
057460 012556          .WORD  TST5+2      ;;STARTING ADDRESS OF TEST 5
057462 013516          .WORD  TST6+2      ;;STARTING ADDRESS OF TEST 6
057464 014336          .WORD  TST7+2      ;;STARTING ADDRESS OF TEST 7
057466 015126          .WORD  TST10+2     ;;STARTING ADDRESS OF TEST 10
057470 016064          .WORD  TST11+2     ;;STARTING ADDRESS OF TEST 11
057472 016660          .WORD  TST12+2     ;;STARTING ADDRESS OF TEST 12

```

```

057474 017472 .WORD TST13+2      ;; STARTING ADDRESS OF TEST 13
057476 020304 .WORD TST14+2      ;; STARTING ADDRESS OF TEST 14
057500 021126 .WORD TST15+2      ;; STARTING ADDRESS OF TEST 15
057502 021670 .WORD TST16+2      ;; STARTING ADDRESS OF TEST 16
057504 022432 .WORD TST17+2      ;; STARTING ADDRESS OF TEST 17
057506 023174 .WORD TST20+2      ;; STARTING ADDRESS OF TEST 20
057510 023500 .WORD TST21+2      ;; STARTING ADDRESS OF TEST 21
057512 024004 .WORD TST22+2      ;; STARTING ADDRESS OF TEST 22
057514 024346 .WORD TST23+2      ;; STARTING ADDRESS OF TEST 23
057516 024670 .WORD TST24+2      ;; STARTING ADDRESS OF TEST 24
057520 025206 .WORD TST25+2      ;; STARTING ADDRESS OF TEST 25
057522 026120 .WORD TST26+2      ;; STARTING ADDRESS OF TEST 26
057524 026742 .WORD TST27+2      ;; STARTING ADDRESS OF TEST 27
057526 027524 .WORD TST30+2      ;; STARTING ADDRESS OF TEST 30
057530 030674 .WORD TST31+2      ;; STARTING ADDRESS OF TEST 31

2
3
4
5
057532
057532 012746 000140
057536 012746 057544
057542 000002
057544
6
7
8
9
057544 012746 000240
057550 012746 057556
057554 000002
057556
10 057556 000207

;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
STOP:
MOV #PR3,-(SP)      ;;PUT NEW PS ON STACK
MOV #64$,-(SP)     ;;PUT NEW PC ON STACK
RTI                 ;;POP NEW PC AND PS
64$:

;RAISE PRIORITY TO INHIBIT CONSOLE INTERRUPT
MOV #PR5,-(SP)     ;;PUT NEW PS ON STACK
MOV #65$,-(SP)     ;;PUT NEW PC ON STACK
RTI                 ;;POP NEW PC AND PS
65$:
RTS PC              ;RETURN

```

ERROR HANDLER ROUTINE

1

.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 ERRTP 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

057560 105037 060152 $ERROR: CLRB      IBSAVE      ;;CLEAR THE ITEM BYTE SAVE LOCATION
057564 104410          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
057566 105237 001117 7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
057572 001775          BEQ      7$          ;;DON'T LET THE FLAG GO TO ZERO
057574 013777 001116 121354 MOV      $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
057602 032777 002000 121344 BIT      #BIT10,@SWR      ;;BELL ON ERROR?
057610 001402          BEQ      1$          ;;NO - SKIP
057612 104401 001212          TYPE      ,SBELL      ;;RING BELL
057616 005237 001126 1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
057622 011637 001132          MOV      (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
057626 162737 000002 001132 SUB      #2,$ERRPC
057634 117737 121272 001130 MOVB    @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
057642 032777 001000 121304 BIT      #BIT09,@SWR      ;;SEE IF LOOP ON ERROR IS SET
057650 001060          BNE      1004$      ;;BRANCH AROUND ROUTINE IF SO
057652 122737 000177 001130 CMPB    #177,$ITEMB      ;;SEE IF THIS IS THE POWER FAIL CALL
057660 001454          BEQ      1004$      ;;BRANCH AROUND ROUTINE IF IT IS
057662 105737 060152          TSTB    IBSAVE      ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
057666 001047          BNE      1003$      ;;BRANCH IF SO
057670 022737 177777 060150 CMP      #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
057676 001445          BEQ      1004$      ;;BRANCH IF SO
057700 013746 000004          MOV      ERRVEC,-(SP) ;;SAVE CONTENTS OF ERROR VECTOR
057704 012737 057722 000004 MOV      #1000$,ERRVEC ;;SETUP 'TRAP' RETURN ADDRESS
057712 013737 177766 060150 MOV      177766,CPSAVE ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
057720 000406          BR      1001$
057722 012737 177777 060150 1000$: MOV      #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
057730 012716 057736          MOV      #1001$,(SP) ;;SETUP RETURN ADDRESS
057734 000002          RTI
057736 012637 000004          1001$: MOV      (SP)+,ERRVEC ;;RESTORE CONTENTS OF ERROR VECTOR

057742 022737 177777 060150 1002$: CMP      #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
057750 001420          BEQ      1004$      ;;BRANCH IF SO
057752 032737 000001 060150 BIT      #BIT00,CPSAVE      ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
057760 001414          BEQ      1004$      ;;BRANCH IF OK
057762 042737 000001 177766 BIC      #BIT00,177766 ;;CLEAR THE BIT FOUND SET
057770 113737 001130 060152 MOVB    $ITEMB,IBSAVE ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
057776 112737 000177 001130 MOVB    #177,$ITEMB      ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
060004 000402          BR      1004$      ;;BRANCH OVER IBSAVE CLEARING

060006 105037 060152          1003$: CLRB      IBSAVE      ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
060012          1004$:
060012 032777 020000 121134 BIT      #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
060020 001004          BNE      20$          ;;SKIP TYPEOUTS
060022 004737 060154          JSR      PC,ERRTYP ;;GO TO USER ERROR ROUTINE

```

```

060026 104401 001217          TYPE      ,SCLRF
060032          20$:
060032 122737 000001 001242  CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE
060040 001007          BNE      2$              ;;NO,SKIP APT ERROR REPORT
060042 113737 001130 060054  MOVB     $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
060050 004737 063116          JSR      PC,$ATY4        ;;REPORT FATAL ERROR TO APT
060054          21$:
060055          .BYTE    0
060056 000777          .BYTE    0
060060 105737 060152          BR       22$              ;;APT ERROR LOOP
060064 001005          TSTB     IBSAVE           ;;SEE IF IBSAVE IS LOADED
060066 005777 121062          BNE     3$              ;;BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
060072 100002          TST     @SWR            ;;HALT ON ERROR
060074 000000          BPL     3$              ;;SKIP IF CONTINUE
060076 104410          HALT                    ;;HALT ON ERROR!
060100          3$:
060100 032777 001000 121046  CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
060106 001402          BIT     #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
060110 013716 001124          BEQ     4$              ;;BR IF NO
060114 005737 001210          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
060120 001402          TST     $ESCAPE        ;;CHECK FOR AN ESCAPE ADDRESS
060122 013716 001210          BEQ     5$              ;;BR IF NONE
060126          4$:
060126 022737 032122 000042  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
060134 001001          CMP     #SENDAD,@#42   ;;ACT-11 AUTO-ACCEPT?
060136 000000          BNE     6$              ;;BRANCH IF NO
060140          5$:
060140 105737 060152          HALT                    ;;YES
060140 105737 060152          TSTB     IBSAVE           ;;SEE IF ITEM BYTE SAVE LOCATION HAS AN ERROR CALL
060144 001210          BNE     7$              ;;BRANCH BACK TO CALL ORIGINAL ERROR
060146 000002          RTI                    ;;RETURN
060150 000000          CPSAVE: .WORD    0     ;;LOCATION TO SAVE CPU ERROR REG CONTENTS
060152 000000          IBSAVE: .WORD    0     ;;LOCATION TO SAVE ITEM BYTE

```

.SBTTL ERROR TYPEOUT ROUTINE

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

```

; *THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
; *REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
; *
; *   .UNIT NUMBER, DRIVE TYPE, TEST NUMBER, ERROR NUMBER AND
; *PROGRAM COUNTER ARE PRINTED ON THE FIRST LINE;
; *   .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
; *ONE OR MORE SUCCEEDING LINES;
; *   .PAIRED LINES OF ERROR HEADERS AND ERROR DATA ARE PRINTED
; *AFTER THE ERROR MESSAGE.
  
```

```

ERRRYP: SAVREG
          BIT      #SW13,@SWR      ;INHIBIT TYPEOUTS??
          BEQ      1$              ;NO!.
          JMP      2$              ;YES.!

;TYPE UNIT NUMBER, DRIVE TYPE, TEST NUMBER, ERROR NUMBER, AND
;PROGRAM COUNTER
1$:      TYPE     ,SCLRF
          TYPE     ,ERTY00        ;TYPE 'DRV#'
          MOV      $UNIT,-(SP)    ;:SAVE $UNIT FOR TYPEOUT
          TYPOS   ;:TYPE DRIVE NUMBER
          .BYTE   3               ;:GO TYPE--OCTAL ASCII
          .BYTE   0               ;:TYPE 3 DIGIT(S)
          .BYTE   0               ;:SUPPRESS LEADING ZEROS

;TYPE 'DRIVE TYPE' RM05, RM03 OR RM02 FOR UNIT UNDER TEST
          MOV      $BASE,R0       ;GET RM BASE ADDRESS
          MOV      RMDT(R0),R0    ;GET DRIVE TYPE REGISTER
          BIC      #177740,R0     ;SAVE DRIVE TYPE BITS AND
          MOV      #SRM03,3$      ;GET ASCII DRIVE TYPE
          CMP      #24,R0        ;IS DEVICE AN RM03 ?
          BEQ      2$            ;YES !!

          MOV      #SRM02,3$      ;SAVE ASCII DRIVE TYPE
          CMP      #25,R0        ;IS DEVICE AN RM02 ?
          BEQ      2$            ;YES !!

          MOV      #SRM05,3$      ;SAVE ASCII DRIVE TYPE
          CMP      #27,R0        ;IS DEVICE AN RM05 ?
          BNE      4$            ;NO !!
2$:      TYPE     ,ERTY05        ;TYPE '' - ''
          TYPE     ;TYPE DRIVE TYPE
          .WORD    0              ;DRIVE TYPE MESSAGE IS STORED HERE

;TYPE TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER
4$:      CLR      TSTNMB         ;LOAD TEST NUMBER FOR
          MOV      $TESTN,TSTNMB
          TYPE     ,ERTY01        ;TYPE 'TST#'
          MOV      TSTNMB,-(SP)   ;:SAVE TSTNMB FOR TYPEOUT
          TYPOS   ;:TYPE TEST NUMBER
          .BYTE   3               ;:GO TYPE--OCTAL ASCII
          .BYTE   0               ;:TYPE 3 DIGIT(S)
          .BYTE   0               ;:SUPPRESS LEADING ZEROS
          CLR      ERRNMB         ;LOAD ERROR NUMBER FOR
          MOVB    $ITEMB,ERRNMB  ;TYPEOUT
  
```

```

060154 104414
060156 032777 020000 *20770
060164 001402
060166 000137 061014

060172 104401 0C1217
060176 104401 061030
060202 013746 001234

060206 104403
060210 003
060211 000

060212 013700 001276
060216 016000 000026
060222 042700 177740
060226 012737 064373 060300
060234 022700 000024
060240 001414

060242 012737 064366 060300
060250 022700 000025
060254 001406

060256 012737 064400 060300
060264 022700 000027
060270 001004
060272 104401 061065
060276 104401
060300 000000

060302 005037 061020
060306 013737 001226 061020
060314 104401 061035
060320 013746 061020

060324 *04403
060326 003
060327 000

060330 005037 061022
060334 113737 001130 061022
  
```

```

50 060342 001406          BEQ      5$      ;SKIP IF NO ERROR CALLED
51 060344 104401 061045   TYPE     ,ERTY02  ;TYPE 'ERR#'
52 060350 013746 061022   MOV      ERRNMB,-(SP) ;SAVE ERRNMB FOR TYPEOUT
                                ;TYPE ERROR NUMBER
                                ;GO TYPE--OCTAL ASCII
                                ;TYPE 3 DIGIT(S)
                                ;SUPPRESS LEADING ZEROS
                                ;TYPE 'PC='
                                ;SAVE $ERRPC FOR TYPEOUT
                                ;TYPE PROGRAM COUNTER
                                ;GO TYPE--OCTAL ASCII
                                ;TYPE 6 DIGIT(S)
                                ;TYPE LEADING ZEROS
      060354 104403          TYPOS
      060356      003        .BYTE     3
      060357      000        .BYTE     0
53 060360 104401 061054   5$:     TYPE     ,ERTY03
54 060364 013746 001132   MOV      $ERRPC,-(SP)
                                ;TYPE PROGRAM COUNTER
                                ;GO TYPE--OCTAL ASCII
                                ;TYPE 6 DIGIT(S)
                                ;TYPE LEADING ZEROS
      060370 104403          TYPOS
      060372      006        .BYTE     6
      060373      001        .BYTE     1

55
56 ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
57 060374 005737 061022   6$:     TST      ERRNMB ;WAS AN ERROR CALLED?
58 060400 001002          BNE      7$      ;BR IF YES
59 060402 000137 061014   JMP      27$     ;NO--EXIT

60
61 060406 104401 001217   7$:     TYPE     ,SCLF  ;YES-TYPE CRLF
62 060412 105037 061026   CLR      BOTFLG  ;CLEAR BOT FLAG
63 060416 105037 061027   CLR      CHRCNT  ;CLEAR CHARACTER COUNTER
64 060422 013700 061022   MOV      ERRNMB,R0 ;R0 POINTS TO FIRST OF
65 060426 122700 000177   CMPB    #177,R0  ;SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
66 060432 001003          BNE      8$      ;BRANCH IF NOT
67 060434 012700 061072   MOV      #PFECH,R0 ;MOVE POWER FAIL ERROR CALL TABLE TO R0
68 060440 000405          BR
69 060442 006300          8$:     ASL      R0      ;FOUR ENTRIES IN ERROR
70 060444 006300          ASL      R0      ;TABLE
71 060446 006300          ASL      R0
72 060450 062700 001572   ADD     #SERRTB-8.,R0
73 060454 011001          9$:     MOV      (R0),R1 ;R1 POINTS TO ERROR MESSAGE
74 ;TABLE
75 060456 001507          BEQ      19$     ;BRANCH IF NO ERROR MESSAGE
76
77 ;TYPE THE ERROR MESSAGE
78 060460 012102          10$:    MOV      (R1)+,R2 ;R2=ADDRESS OF MESSAGE STRING
79 060462 001505          BEQ      19$     ;BRANCH IF END OF MESSAGE
80 060464 010237 060632   MOV      R2,18$  ;LOAD ADDRESS OF STRING
81 060470 005037 061024   CLR      BOTADR  ;CLEAR BOT ADDRESS
82 060474 112203          11$:    MOV      (R2)+,R3 ;END OF STRING??
83 060476 001454          BEQ      17$     ;YES!!
84 060500 122703 000015   CMPB    #CR,R3  ;CARRIAGE RETURN??
85 060504 001003          BNE      12$     ;NO!!
86 060506 105037 061027   CLR      CHRCNT ;YES-CLEAR CHAR COUNT
87 060512 000770          BR       11$    ;GET NEXT CHARACTER
88 060514 122703 000012   12$:    CMPB    #LF,R3 ;LINE FEED??
89 060520 001765          BEQ      11$     ;YES-GET NEXT CHARACTER
90 060522 122703 000011   CMPB    #HT,R3  ;HORIZONTAL TAB??
91 060526 001007          BNE      14$     ;NO!!
92 060530 105237 061027   13$:    INCB    CHRCNT ;ADJUST CHARACTER COUNT
93 060534 132737 000007 061027   BITB    #7,CHRCNT
94 060542 001372          BNE      13$
95 060544 000407          BR       15$
96 060546 105237 061027   14$:    INCB    CHRCNT ;INCREMENT CHARACTER COUNT
97 060552 122703 000040   CMPB    #' ',R3 ;SPACE??
98 060556 001002          BNE      15$     ;NO!!

```

```

ERROR TYPEOUT ROUTINE
 99 060560 010237 061024      MOV      R2,BOTADR      ;SAVE ADDRESS OF SPACE
100 060564 122737 000100 061027 15$:  CMPS      #64.,CHRCNT    ;END OF LINE??
101 060572 103340          BHIS      11$          ;NO.
102 060574 013704 061024      MOV      BOTADR,R4      ;GET ADDRESS OF LAST SPACE
103 060600 001007          BNE      16$          ;BRANCH IF SPACE DETECTED
104 060602 104401 001217      TYPE     , $CRLF        ;TYPE CRLF
105 060606 105037 061027      CLRB     CHRCNT         ;CLEAR CHARACTER COUNT
106 060612 013702 060632      MOV      18$,R2        ;SET UP R2 FOR TESTING
107 060616 000726          BR       11$
108 060620 105044          CLRB     -(R4)         ;REPLACE SPACE
109 060622 112737 177777 061026 6$:   MOVVB    #-1,BOTFLG     ;SET BOT FLAG
110 060630 104401          TYPE     ;TYPE ERROR MESSAGE STRING
111 060632 000000          .WORD   ;STRING ADDRESS GOES HERE
112 060634 105737 061026 18$:   TSTB     BOTFLG        ;WAS STRING TRUNCATED??
113 060640 001707          BEQ     10$          ;NO!!
114 060642 104401 001217      TYPE     , $CRLF        ;YES-TYPE CRLF
115 060646 105037 061026      CLRB     BOTFLG        ;CLEAR BOT FLAG
116 060652 105037 061027      CLRB     CHRCNT         ;CLEAR CHARACTER COUNT
117 060656 013702 061024      MOV      BOTADR,R2     ;SETUP R2 FOR TESTING
118 060662 010237 060632      MOV      R2,18$        ;SETUP 18$ FOR TYPING
119 060666 112742 000040      MOVVB    #'-,(R2)      ;RESTORE SPACE
120 060672 105722          TSTB     (R2)+         ;RESTORE R2
121 060674 000677          BR       11$          ;TYPE REST OF STRING
122
123          ;TYPE ERROR HEADER AND ERROR DATA
124 060676 016001 000002 19$:   MOV      2(R0),R1      ;R1 POINTS TO ERROR HEADER TABLE
125 060702 001444          BEQ     27$          ;BRANCH IF NO HEADER
126 060704 104401 001217      TYPE     , $CRLF        ;(ASSUME NO DATA)
127 060710 016002 000004      MOV      4(R0),R2      ;R2 POINTS TO DATA ADDRESS TABLE
128 060714 016003 000006      MOV      6(R0),R3      ;R3 POINTS TO FORMAT TABLE
129 060720 012137 060730 20$:   MOV      (R1)+,21$     ;PUT HEADER ADDRESS FOR TYPE
130 060724 001433          BEQ     27$          ;BRANCH IF END OF HEADERS
131          ;(ASSUME END OF DATA)
132 060726 104401          TYPE     ;HEADER ADDRESS GOES HERE
133 060730 000000 21$:   .WORD   0
134 060732 104401 001217      TYPE     , $CRLF
135 060736 005702          TST     R2            ;DATA WITH HEADER??
136 060740 001767          BEQ     20$          ;NO!!
137 060742 012204          MOV     (R2)+,R4      ;R4 POINTS TO DATA ADDRESS
138 060744 012305          MOV     (R3)+,R5      ;R5 POINTS TO FORMAT
139 060746 105725 22$:   TSTB     (R5)+         ;WHAT KIND OF DATA??
140 060750 100407          BMI     24$          ;BINARY
141 060752 001403          BEQ     23$          ;OCTAL
142 060754 013446          MOV     @ (R4)+,-(SP) ;DECIMAL
143 060756 104405          TYPDS
144 060760 000405          BR     25$
145 060762 013446 23$:   MOV     @ (R4)+,-(SP)
146 060764 104402          TYPOC
147 060766 000402          BR     25$
148 060770 013446 24$:   MOV     @ (R4)+,-(SP)
149 060772 104406          TYPBN
150 060774 005714 25$:   TST     (R4)          ;MORE DATA??
151 060776 001403          BEQ     26$          ;NO.
152 061000 104401 061062      TYPE     ,ERTY04       ;YES-TYPE 2 SPACES
153 061004 000760          BR     22$          ;AND CONTINUE
154 061006 104401 001217 26$:   TYPE     , $CRLF        ;TYPE ONE BLANK LINE
155 061012 000742          BR     20$          ;BEFORE NEXT HEADER

```

156 061014 104415
157 061016 000207
158
159 061020 000000
160 061022 000000
161 061024 000000
162 061026 000
163 061027 000
164
165 061030 104 122
166 061035 054 040
167 061045 054 040
168 061054 054 040
169 061062 040 040
170 061065 040 055
171
172 061072 061102 061170 061206
173 061102 061106 000000
174 061106 120 117 127
175
176 061170 061174 000000
177 061174 103 120 125
178
179 061206 061210
180 061210 060150 000000
181 061214 061216
182 061216 000 000

275: RESREG
RTS PC
TSTNMB: .WORD 0 :TEST NUMBER
ERRNMB: .WORD 0 :ERROR NUMBER
BOTADR: .WORD 0 :BEGINNING OF TEXT ADDRESS
BOTFLG: .BYTE 0 :BOT FLAG
CHRCNT: .BYTE 0 :CHARACTER COUNT
ERTY00: .ASCIZ @DRV#@
ERTY01: .ASCIZ @, TEST#@
ERTY02: .ASCIZ @, ERR#@
ERTY03: .ASCIZ @, PC=@
ERTY04: .ASCIZ @ @
ERTY05: .ASCIZ @ - @
.EVEN
PFECH: PFECH1,PFECH2,PFECH3,PFECH4 ;WORDS DEFINING TABLES BELOW
PFECH1: .+4,0
.ASCIZ ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?
.EVEN
PFECH2: .+4,0
.ASCIZ ?CPUERREG?
.EVEN
PFECH3: .+2
.WORD (PSAVE,0
PFECH4: .+2
.BYTE 0,0

.SBTTL TTY INPUT ROUTINE

```

*****
ENABL  LSB
061220 000000 $TKCNT: .WORD 0      ;;NUMBER OF ITEMS IN QUEUE
061222 000000 $TKQIN: .WORD 0      ;;INPUT POINTER
061224 000000 $TKQOUT: .WORD 0     ;;OUTPUT POINTER
061226 061227 $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
;
061230 005037 061220 $TKINT: CLR      $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
061234 012737 061226 061222 MOV      # $TKQSRV,$TKQIN  ;;MOVE THE STARTING ADDRESS OF THE
061242 013737 061222 061224 MOV      $TKQIN,$TKQOUT    ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
061250 012737 061300 000060 MOV      # $TKSRV,@ $TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
061256 012737 000200 000062 MOV      #200,@ $TKVEC+2   ;;'BR' LEVEL 4
061264 005777 117672 *ST      @ $TKB           ;;CLEAR DONE FLAG
061270 012777 000100 117662 MOV      #100,@ $TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
061276 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 (SHUT2)
;
061300 117746 117656 $TKSRV: MOVB     @ $TKB,-(SP)  ;;PICKUP THE CHARACTER
061304 042716 177600 BIC      #^C(177),(SP)     ;;STRIP THE JUNK
061310 021627 000021 CMP      (SP),#$XON       ;;IS IT A RANDOM XON?
061314 001002 BNE      30$             ;;BRANCH IF NO
061316 005726 TST      (SP)+           ;;CLEAN RANDOM XON OFF STACK
061320 000002 RTI                          ;;RETURN
061322 30$:
061322 021627 000003 CMP      (SP),#3          ;;IS IT A CONTROL C?
061326 001007 BNE      1$             ;;BRANCH IF NO
061330 104401 062426 TYPE     , $CNTLC          ;;TYPE A CONTROL-C (^C)
061334 004737 061230 JSR      PC,$TKINT      ;;INIT THE KEYBOARD
061340 005726 TST      (SP)+           ;;CLEAN UP STACK
061342 000137 062470 JMP      SHUT2          ;;CONTROL C RESTART
061346 021627 000007 1$: CMP      (SP),#7          ;;IS IT A CONTROL G?
061352 001004 BNE      2$             ;;BRANCH IF NO
061354 022737 000176 001154 CMP      #SWREG,SWR     ;;IS SOFT-SWR SELECTED?
061362 001500 BEQ      6$             ;;GO TO SWR CHANGE

061364 2$:
061364 022737 000001 061220 CMP      #1,$TKCNT      ;;IS THE QUEUE FULL?
061372 001004 BNE      3$             ;;BRANCH IF NO
061374 104401 001212 TYPE     , $BELL          ;;RING THE TTY BELL

```

```

061400 005726          TST      (SP)+          ;;CLEAN CHARACTER OFF OF STACK
061402 000451          BR       5$          ;;EXIT
061404 021627 000023 3$:      CMP      (SP),#23          ;;IS IT A CONTROL-S?
061410 001021          BNE     32$          ;;BRANCH IF NO
061412 005077 117542      CLR     @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
061416 005726          TST      (SP)+          ;;CLEAN CHAR OFF STACK
061420 105777 117534 31$:    TSTB   @STKS          ;;WAIT FOR A CHAR
061424 100375          BPL     31$          ;;LOOP UNTIL ITS THERE
061426 117746 117530      MOVB   @STKB,-(SP)    ;;GET THE CHARACTER
061432 042716 177600      BIC     #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
061436 022627 000021      CMP     (SP)+,#21    ;;IS IT A CONTROL-Q?
061442 001366          BNE     31$          ;;BRANCH IF NO
061444 012777 000100 117506      MOV     #100,@STKS   ;;REENABLE TTY KEYBOARD INTERRUPTS
061452 000002          RTI                    ;;RETURN
061454 005237 061220 32$:    INC     $TKCNT        ;;COUNT THIS CHARACTER
061460 021627 000140      CMP     (SP),#140    ;;IS IT UPPER CASE?
061464 002405          BLT     4$          ;;BRANCH IF YES
061466 021627 000175      CMP     (SP),#175    ;;IS IT A SPECIAL CHAR?
061472 003002          BGT     4$          ;;BRANCH IF YES
061474 042716 0C0040      BIC     #40,(SP)     ;;MAKE IT UPPER CASE
061500 112677 177516 4$:      MOVB   (SP)+,@STKQIN  ;;AND PUT IT IN QUEUE
061504 005237 061222      INC     $TKQIN        ;;UPDATE THE POINTER
061510 023727 061222 061227      CMP     $TKQIN,$$TKQEND ;;GO OFF THE END?
061516 001003          BNE     5$          ;;BRANCH IF NO
061520 012737 061226 061222      MOV     $$TKQRT,$$TKQIN ;;RESET THE POINTER
061526 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.

```

061530 022737 000176 001154 $CKSWR: CMP     #SWREG,$SWR    ;;IS THE SOFT-SWR SELECTED
061536 001124          BNE     15$          ;;EXIT IF NOT
061540 105777 117414      TSTB   @STKS          ;;IS A CHAR WAITING?
061544 100121          BPL     15$          ;;IF NOT, EXIT
061546 117746 117410      MOVB   @STKB,-(SP)    ;;YES
061552 042716 177600      BIC     #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
061556 021627 000007      CMP     (SP),#7      ;;IS IT A CONTROL-G?
061562 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.

```

061564 123727 001150 000001 6$:      CMPS   $AUTOB,#1     ;;ARE WE RUNNING IN AUTO-MODE?
061572 001674          BEQ     2$          ;;BRANCH IF YES
061574 005726          TST     (SP)+          ;;CLEAR CONTROL-G OFF STACK
061576 004737 061230      JSR    PC,$TKINT     ;;FLUSH THE TTY INPUT QUEUE
061602 005077 117352      CLR     @STKS          ;;DISABLE TTY KEYBOARD INTERRUPTS
061606 112737 000001 001151      MOVB   #1,$INTAG     ;;SET INTERRUPT MODE INDICATOR

061614 104401 062440      TYPE   ,SCNTLG        ;;ECHO THE CONTROL-G (^G)
061620 104401 062445      $GTSWR: TYPE   ,SMSWR    ;;TYPE CURRENT CONTENTS
061624 013746 000176      MOV     SWREG,-(SP)   ;;SAVE SWREG FOR TYPEOUT
061630 104402          TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)

```

```

061632 104401 062456          TYPE      ,SMNEW          ;;PROMPT FOR NEW SWR
061636 005046          CLR        -(SP)          ;;CLEAR COUNTER
061640 005046          CLR        -(SP)          ;;THE NEW SWR
061642 105777 117312      7$:      TSTB      @STKS          ;;CHAR THERE?
061646 100375          BPL        7$            ;;IF NOT TRY AGAIN

061650 117746 117306      MOVB      @STKB,-(SP)      ;;PICK UP CHAR
061654 042716 177600      BIC      #^C17,(SP)     ;;MAKE IT 7-BIT ASCII

061660 021627 000003      CMP      (SP),#3         ;;IS IT A CONTROL-C?
061664 001015          BNE      9$             ;;BRANCH IF NOT
061666 104401 062426      TYPE      ,SCNTLC       ;;YES, ECHO CONTROL-C (^C)
061672 062706 000006      ADD      #6,SP          ;;CLEAN UP STACK
061676 123727 001151 000001  CMPB      $INTAG,#1      ;;REENABLE TTY KEYBOARD INTERRUPTS?
061704 001003          BNE      8$             ;;BRANCH IF NO
061706 012777 000100 117244  MOV      #100,@STKS     ;;ALLOW TTY KEYBOARD INTERRUPTS
061714 000137 062470      8$:      JMP      SHUT2        ;;CONTROL-C RESTART

061720 021627 000025      9$:      CMP      (SP),#25      ;;IS IT A CONTROL-U?
061724 001005          BNE      10$            ;;BRANCH IF NOT
061726 104401 062433      TYPE      ,SCNTLU       ;;YES, ECHO CONTROL-U (^U)
061732 062706 000006      20$:     ADD      #6,SP          ;;IGNORE PREVIOUS INPUT
061736 000737          BR       19$            ;;LET'S TRY IT AGAIN

061740 021627 000015      10$:     CMP      (SP),#15      ;;IS IT A <CR>?
061744 001022          BNE      16$            ;;BRANCH IF NO
061746 005766 000004      TST      4(SP)          ;;YES, IS IT THE FIRST CHAR?
061752 001403          BEQ      11$            ;;BRANCH IF YES
061754 016677 000002 117172  MOV      2(SP),@SWR     ;;SAVE NEW SWR
061762 062706 000006      11$:     ADD      #6,SP          ;;CLEAN UP STACK
061766 104401 001217      14$:     TYPE      ,SCRLF       ;;ECHO <CR> AND <LF>
061772 123727 001151 000001  CMPB      $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
062000 001003          BNE      15$            ;;BRANCH IF NOT
062002 012777 000100 117150  MOV      #100,@STKS     ;;RE-ENABLE TTY KBD INTERRUPTS
062010 000002          RTI                      ;;RETURN
062012 004737 056642      15$:     JSR      PC,$TYPEC     ;;ECHO CHAR
062016 021627 000060      16$:     CMP      (SP),#60      ;;CHAR < 0?
062022 002420          RLT      18$            ;;BRANCH IF YES
062024 021627 000067      CMP      (SP),#67      ;;CHAR > 7?
062030 003015          BGT      18$            ;;BRANCH IF YES
062032 042726 000060      BIC      #60,(SP)+      ;;STRIP-OFF ASCII
062036 005766 000002      TST      2(SP)          ;;IS THIS THE FIRST CHAR
062042 001403          BEQ      17$            ;;BRANCH IF YES
062044 006316          ASL      (SP)          ;;NO, SHIFT PRESENT
062046 006316          ASL      (SP)          ;;CHAR OVER TO MAKE
062050 006316          ASL      (SP)          ;;ROOM FOR NEW ONE.
062052 005266 000002      17$:     INC      2(SP)          ;;KEEP COUNT OF CHAR
062056 056616 177776      BIS      -2(SP),(SP)    ;;SET IN NEW CHAR
062062 000667          BR      7$             ;;GET THE NEXT ONE
062064 104401 001216      18$:     TYPE      ,SQUES     ;;TYPE ?<CR><LF>
062070 000720          BR      20$            ;;SIMULATE CONTROL-U
.DSABL  LSB

```

```

: *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
: *CALL:
: *      RDCHR          ;; GET A CHARACTER FROM THE QUEUE
: *      RETURN HERE   ;; CHARACTER IS ON THE STACK
: *                  ;; WITH PARITY BIT STRIPPED OFF
:
062072 011646          $RDCHR: MOV      (SP),-(SP)      ;; PUSH DOWN THE PC AND
062074 016666 000004 000002 MOV      4(SP),2(SP)      ;; THE PS
062102 005066 000004          CLR      4(SP)          ;; GET READY FOR A CHARACTER
062106 005046          CLR      -(SP)          ;; PUT NEW PS ON STACK
062110 012746 062116          MOV      #64$,-(SP)      ;; PUT NEW PC ON STACK
062114 000002          RTI          ;; POP NEW PC AND PS
062116
062116 005737 061220 64$:   TST      $TKCNT          ;; WAIT ON A CHARACTER
062122 001775          BEQ      1$
062124 005337 061220          DEC      $TKCNT          ;; DECREMENT THE COUNTER
062130 117766 177070 000004 MOVVB   @$TKQOUT,4(SP)      ;; GET ONE CHARACTER
062136 005237 061224          INC      $TKQOUT          ;; UPDATE THE POINTER
062142 023727 061224 061227 CMP      $TKQOUT,#$TKQEND ;; DID IT GO OFF OF THE END?
062150 001003          BNE      2$
062152 012737 061226 061224 MOV      #$TKQSRST,$TKQOUT ;; RESET THE POINTER
062160 000002          RTI          ;; RETURN
: *****
: *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
: *CALL:
: *      RDLIN         ;; INPUT A STRING FROM THE TTY
: *      RETURN HERE   ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
: *                  ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
:
062162 010346          $RDLIN: MOV      R3, -(SP)          ;; SAVE R3
062164 005046          CLR      -(SP)          ;; CLEAR THE RUBOUT KEY
062166 012703 062416 1$:   MOV      #$TTYIN,R3          ;; GET ADDRESS
062172 022703 062426 2$:   CMP      #$TTYIN+8.,R3      ;; BUFFER FULL?
062176 101456          BLOS     4$
062200 104411          RDCHR          ;; GO READ ONE CHARACTER FROM THE TTY
062202 112613          MOVVB   (SP)+,(R3)          ;; GET CHARACTER
062204 122713 000177 10$:  CMPB    #177,(R3)          ;; IS IT A RUBOUT
062210 001022          BNE      5$
062212 005716          TST      (SP)          ;; IS THIS THE FIRST RUBOUT?
062214 001007          BNE      6$
062216 112737 000134 062414 MOVVB   #' \,9$          ;; TYPE A BACK SLASH
062224 104401 062414          TYPE     ,9$
062230 012716 177777          MOV      #-1,(SF)          ;; SET THE RUBOUT KEY
062234 005303          DEC      R3          ;; BACKUP BY ONE
062236 020327 062416 6$:   CMP      R3,#$TTYIN      ;; STACK EMPTY?
062242 103434          BLO      4$
062244 111337 062414          MOVVB   (R3),9$          ;; SETUP TO TYPEOUT THE DELETED CHAR.
062250 104401 062414          TYPE     ,9$
062254 000746          BR       2$
062256 005716          5$:   TST      (SP)          ;; RUBOUT KEY SET?
062260 001406          BEQ      7$
062262 112737 000134 062414 MOVVB   #' \,9$          ;; TYPE A BACK SLASH
062270 104401 062414          TYPE     ,9$
062274 005016          CLR      (SP)          ;; CLEAR THE RUBOUT KEY
062276 122713 000025 7$:   CMPB    #25,(R3)          ;; IS CHARACTER A CTRL U?
062302 001003          BNE      8$
062302          BR       8$

```

TTY INPUT ROUTINE

062304	104401	062433			TYPE	,\$CNTLU	::TYPE A CONTROL 'U'	
062310	000726				BR	1\$::GO START OVER	
062312	122713	000022		8\$:	CMPB	#22,(R3)	::IS CHARACTER A '^R'?	
062316	001011				BNE	3\$::BRANCH IF NO	
062320	105013				CLRB	(R3)	::CLEAR THE CHARACTER	
062322	104401	001217			TYPE	,\$CRLF	::TYPE A 'CR' & 'LF'	
062326	104401	062416			TYPE	,\$TTYIN	::TYPE THE INPUT STRING	
062332	000717				BR	2\$::GO PICKUP ANOTHER CHACTER	
062334	104401	001216		4\$:	TYPE	,\$QUES	::TYPE A '?'	
062340	000712				BR	1\$::CLEAR THE BUFFER AND LOOP	
062342	111337	062414		3\$:	MOVB	(R3),9\$::ECHO THE CHARACTER	
062346	104401	062414			TYPE	,\$		
062352	122723	000015			CMPB	#15,(R3)+	::CHECK FOR RETURN	
062356	001305				BNE	2\$::LOOP IF NOT RETURN	
062360	105063	177777			CLRB	-1(R3)	::CLEAR RETURN (THE 15)	
062364	104401	001220			TYPE	,\$LF	::TYPE A LINE FEED	
062370	005726				TST	(SP)+	::CLEAN RUBOUT KEY FROM THE STACK	
062372	012603				MOV	(SP)+,R3	::RESTORE R3	
062374	011646				MOV	(SP)-,(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE	
062376	016666	000004	000002		MOV	4(SP),2(SP)	::FIRST ASCII CHARACTER ON IT	
062404	012766	062416	000004		MOV	#TTYIN,4(SP)		
062412	000002				RTI		::RETURN	
062414	000			9\$:	.BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE	
062415	000				.BYTE	0	::TERMINATOR	
062416					,\$TTYIN:	.BLKB	8.	
062426	136	103	015		,\$CNTLC:	.ASCIZ	/^C/<15><12>	
062433	136	125	015		,\$CNTLU:	.ASCIZ	/^U/<15><12>	
062440	136	107	015		,\$CNTLG:	.ASCIZ	/^G/<15><12>	
062445	015	012	123		,\$MSWR:	.ASCIZ	<15><12>/SWR = /	
062456	040	040	116		,\$MNEW:	.ASCIZ	/ NEW = /	
					.EVEN			
2								
3	062470	012737	177777	001326	SHUT2:	MOV	#-1,CTLFG	::SET THE CONTROL-C FLAG
4	062476	105737	001116			TSTB	,\$STNM	::DOING ANY TESTS ?
5	062502	001002				BNE	1\$::BR IF YES
6	062504	000137	032142			JMP	SHUT	::NO--RESPOND TO ^C
7	062510	000002			1\$:	RTI		::EXIT FROM INTERRUPT

1

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

 *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 *CHANGE IT TO BINARY.

*CALL:
 * RDOCT ;:READ AN OCTAL NUMBER
 * RETURN HERE ;:LOW ORDER BITS ARE ON TOP OF THE STACK
 * ;:HIGH ORDER BITS ARE IN \$HIOCT

062512	011646			\$RDOCT: MOV	(SP),-(SP)	:::PROVIDE SPACE FOR THE
062514	016666	000004	000002	MOV	4(SP),2(SP)	:::INPUT NUMBER
062522	010046			MOV	R0,-(SP)	:::PUSH R0 ON STACK
062524	010146			MOV	R1,-(SP)	:::PUSH R1 ON STACK
062526	010246			MOV	R2,-(SP)	:::PUSH R2 ON STACK
062530	104412			1\$: RDLIN		:::READ AN ASCII LINE
062532	012600			MOV	(SP)+,R0	:::GET ADDRESS OF 1ST CHARACTER
062534	005001			CLR	R1	:::CLEAR DATA WORD
062536	005002			CLR	R2	
062540	112046			2\$: MOVB	(R0)+,-(SP)	:::PICKUP THIS CHARACTER
062542	001412			BEQ	3\$:::IF ZERO GET OUT
062544	006301			ASL	R1	:::*2
062546	006102			ROL	R2	
062550	006301			ASL	R1	:::*4
062552	006102			ROL	R2	
062554	006301			ASL	R1	:::*8
062556	006102			ROL	R2	
062560	042716	177770		BIC	#^C7,(SP)	:::STRIP THE ASCII JUNK
062564	062601			ADD	(SP)+,R1	:::ADD IN THIS DIGIT
062566	000764			BR	2\$:::LOOP
062570	005726			3\$: TST	(SP)+	:::CLEAN TERMINATOR FROM STACK
062572	010166	000012		MOV	R1,12(SP)	:::SAVE THE RESULT
062576	010237	062612		MOV	R2,\$HIOCT	
062602	012602			MOV	(SP)+,R2	:::POP STACK INTO R2
062604	012601			MOV	(SP)+,R1	:::POP STACK INTO R1
062606	012600			MOV	(SP)+,R0	:::POP STACK INTO R0
062610	000002			RTI		:::RETURN
062612	000000			\$HIOCT: .WORD	0	:::HIGH ORDER BITS GO HERE

.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.
    
```

062614	016646	000002	\$TRAP:	MOV	2(SP),-(SP)	::ASSUME THE STATUS OF
062620	042716	000020		BIC	#20,(SP)	:: THE CALLER--DO NOT ALLOW
062624	012746	062632		MOV	#1\$,-(SP)	:: T-BIT TRAPS
062630	000002			RTI		::SET THE NEW STATUS
062632	010046		1\$:	MOV	R0,-(SP)	::SAVE R0
062634	016600	000002		MOV	2(SP),R0	::GET TRAP ADDRESS
062640	005740			TST	-(R0)	::BACKUP BY 2
062642	111000			MOVB	(R0),R0	::GET RIGHT BYTE OF TRAP
062644	006300			ASL	R0	::POSITION FOR INDEXING
062646	016000	062666		MOV	\$TRPAD(R0),R0	::INDEX TO TABLE
062652	000200			RTS	R0	::GO TO ROUTINE

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

062654	011646		\$TRAP2:	MOV	(SP),-(SP)	::MOVE THE PC DOWN
062656	016666	000004		MOV	4(SP),2(SP)	::MOVE THE PSW DOWN
062664	000002	000002		RTI		::RESTORE THE PSW

.SBTTL TRAP TABLE

```

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

			:	ROUTINE		
			:	-----		
062666	062654		\$TRPAD:	.WORD	\$TRAP2	
062670	056430			\$TYPE	::CALL=1TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
062672	056226			\$TYPOC	::CALL=TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
062674	056202			\$TYPOS	::CALL=TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
062676	056242			\$TYPON	::CALL=TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
062700	055756			\$TYPDS	::CALL=TYPDS	TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
062702	055702			\$TYPBN	::CALL=TYPBN	TRAP+6(104406) TYPE BINARY (ASCII) NUMBER
062704	061620			\$GTSWR	::CALL=GTSWR	TRAP+7(104407) GET SOFT-SWR SETTING
062706	061530			\$CKSWR	::CALL=CKSWR	TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
062710	062072			\$RDCHR	::CALL=RDCHR	TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
062712	062162			\$RDLIN	::CALL=RDLIN	TRAP+12(104412) TTY TYPEIN STRING ROUTINE
062714	062512			\$RDOCT	::CALL=RDOCT	TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
062716	055606			\$SAVREG	::CALL=SAVREG	TRAP+14(104414) SAVE R0-R5 ROUTINE
062720	055644			\$RESREG	::CALL=RESREG	TRAP+15(104415) RESTORE R0-R5 ROUTINE

.SBTTL POWER DOWN AND UP ROUTINES

```

*****
:POWER DOWN ROUTINE
062722 012737 063062 000024 $PWRDN: MOV    #SILLUP,@#PWRVEC  ;;SET FOR FAST UP
062730 012737 000340 000026      MOV    #340,@#PWRVEC+2  ;;PRIO:7
062736 010046      MOV    R0,-(SP)        ;;PUSH R0 ON STACK
062740 010146      MOV    R1,-(SP)        ;;PUSH R1 ON STACK
062742 010246      MOV    R2,-(SP)        ;;PUSH R2 ON STACK
062744 010346      MOV    R3,-(SP)        ;;PUSH R3 ON STACK
062746 010446      MOV    R4,-(SP)        ;;PUSH R4 ON STACK
062750 010546      MOV    R5,-(SP)        ;;PUSH R5 ON STACK
062752 017746 116176      MOV    @SWR,-(SP)      ;;PUSH @SWR ON STACK
062756 010637 063066      MOV    SP,$SAVR6      ;;SAVE SP
062762 012737 062774 000024      MOV    #SPWRUP,@#PWRVEC ;;SET UP VECTOR
062770 000000      HALT
062772 000776      BR     .-2            ;;HANG UP

*****
:POWER UP ROUTINE
062774 012737 063062 000024 $PWRUP: MOV    #SILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
063002 013706 063066      MOV    $SAVR6,SP      ;;GET SP
063006 005037 063066      CLR    $SAVR6         ;;WAIT LOOP FOR THE TTY
063012 005237 063066 1$: INC    $SAVR6        ;;WAIT FOR THE INC
063016 001375      BNE   1$              ;;OF WORD
063020 012677 116130      MOV    (SP)+,@SWR     ;;POP STACK INTO @SWR
063024 012605      MOV    (SP)+,R5      ;;POP STACK INTO R5
063026 012604      MOV    (SP)+,R4      ;;POP STACK INTO R4
063030 012603      MOV    (SP)+,R3      ;;POP STACK INTO R3
063032 012602      MOV    (SP)+,R2      ;;POP STACK INTO R2
063034 012601      MOV    (SP)+,R1      ;;POP STACK INTO R1
063036 012600      MOV    (SP)+,R0      ;;POP STACK INTO R0
063040 012737 062722 000024      MOV    #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
063046 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
063054 104401      TYPE                                ;;REPORT THE POWER FAILURE
063056 063070 $PWRMG: .WORD $POWER          ;;POWER FAIL MESSAGE POINTER
063060 000002      RTI
063062 000000 $SILLUP: HALT
063064 000776      BR     .-2            ;;THE POWER UP SEQUENCE WAS STARTED
063066 000000      $SAVR6: 0              ;;BEFORE THE POWER DOWN WAS COMPLETE
063070 015 012 120 $POWER: .ASCIZ <15><12>'POWER' ;;PUT THE SP HERE
          .EVEN

```

2

.SBTTL APT COMMUNICATIONS ROUTINE

```

*****
$ATY1: MOV    #1,$FFLG      ;;TO REPORT FATAL ERROR
063106 112737 000001 063342 $ATY3: MOV    #1,$MFLG      ;;TO TYPE A MESSAGE
063114 000403      BR    $ATYC
063116 112737 000001 063344 $ATY4: MOV    #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
063124 $ATYC:
063124 010046      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
063126 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
063130 105737 063342      TSTB  $MFLG          ;;SHOULD TYPE A MESSAGE?
063134 001450      BEQ   5$             ;;IF NOT: BR
063136 122737 000001 001242      CMPB  #APTENV,$ENV    ;;OPERATING UNDER APT?
063144 001031      BNE   3$             ;;IF NOT: BR
063146 132737 000100 001243      BITB  #APTPOOL,$ENVM  ;;SHOULD SPOOL MESSAGES?

```


063154	001425			BEQ	3\$::IF NOT: BR
063156	017600	000004		MOV	@4(SP),R0	::GET MESSAGE ADDR.
063162	062766	000002	000004	ADD	#2,4(SP)	::BUMP RETURN ADDR.
063170	005737	001222	1\$:	TST	\$MSGTYPE	::SEE IF DONE W/ LAST XMISSION?
063174	001375			BNE	1\$::IF NOT: WAIT
063176	010037	001236		MOV	R0,\$MSGAD	::PUT ADDR IN MAILBOX
063202	105720		2\$:	TSTB	(R0)+	::FIND END OF MESSAGE
063204	001376			BNE	2\$	
063206	163700	001236		SUB	\$MSGAD,R0	::SUB START OF MESSAGE
063212	006200			ASR	R0	::GET MESSAGE LNTH IN WORDS
063214	010037	001240		MOV	R0,\$MSGLGT	::PUT LENGTH IN MAILBOX
063220	012737	000004	001222	MOV	#4,\$MSGTYPE	::TELL APT TO TAKE MSG.
063226	000413			BR	5\$	
063230	017637	000004	063254	3\$:	MOV @4(SP),4\$::PUT MSG ADDR IN JSR LINKAGE
063236	062766	000002	000004	ADD	#2,4(SP)	::BUMP RETURN ADDRESS
063244	013746	177776		MOV	177776,-(SP)	::PUSH 177776 ON STACK
063250	004737	056430		JSR	PC,\$TYPE	::CALL TYPE MACRO
063254	000000		4\$:	.WORD	0	
063256	105737	063344	5\$:			
063262	001416		10\$:	TSTB	\$FFLG	::SHOULD REPORT FATAL ERROR?
063264	005737	001242		BEQ	12\$::IF NOT: BR
063270	001413			TST	\$ENV	::RUNNING UNDER APT?
063272	005737	001222	11\$:	BEQ	12\$::IF NOT: BR
063276	001375			TST	\$MSGTYPE	::FINISHED LAST MESSAGE?
063300	017637	000004	001224	BNE	11\$::IF NOT: WAIT
063306	062766	000002	000004	MOV	@4(SP),\$FATAL	::GET ERROR #
063314	005237	001222		ADD	#2,4(SP)	::BUMP RETURN ADDR.
063320	105037	063344	12\$:	INC	\$MSGTYPE	::TELL APT TO TAKE ERROR
063324	105037	063343		CLRB	\$FFLG	::CLEAR FATAL FLAG
063330	105037	063342		CLRB	\$LFLG	::CLEAR LOG FLAG
063334	012601			CLRB	\$MFLG	::CLEAR MESSAGE FLAG
063336	012600			MOV	(SP)+,R1	::POP STACK INTO R1
063340	000207			MOV	(SP)+,R0	::POP STACK INTO R0
063342	000			RTS	PC	::RETURN
063343	000				0	::MESSG. FLAG
063344	000				0	::LOG FLAG
					0	::FATAL FLAG
					.EVEN	
	000200			APTSIZE	= 200	
	000001			APTENV	001	
	000100			APTSPOOL	100	
	000040			APTSUP	040	

```

        .SBTTL  CONSOLE MESSAGES
2
3 063346      200      103      101  SCTMSG: .ASCIZ  <CRLF>@CANNOT RECOVER THE BAD SECTOR FILES ON THIS DEVICE@
4 063432      075      000      EQUALS: .ASCIZ  @=@
5 063434      101      114      114  ALL: .ASCIZ  @ALL@<CRLF>
6 063441      040      077      040  QUES: .ASCIZ  @ ? @
7 063445      054      040      000  COMMA: .ASCIZ  @, @
8 063450      200      124      117  MSHELP: .ASCII  <CRLF>@TO ENSURE THAT NO BAD HEADERS ARE LEFT ON THE DISK@
9 063536      200      120      101      .ASCII  <CRLF>@PACK, THIS PROGRAM SHOULD BE HALTED BY TYPING A (^C)@
10 063624      200      103      117      .ASCII  <CRLF>@CONTROL C. AS A RESULT, THE PROGRAM WILL BE HALTED@
11 063712      200      127      110      .ASCII  <CRLF>@WHEN THE DRIVE UNDER TEST HAS COMPLETED TESTING.@<CRLF>
12 063774      200      124      131      .ASCIZ  <CRLF>@TYPE HELP TEXT (L) N ? @
13 064025      200      122      115  CNSLO1: .ASCIZ  <CRLF>@RMCS1=@
14 064035      040      114      111  CNSLO2: .ASCIZ  @ LIMITS - LO= 160000, HI= 17XXXX@<CRLF>
15 064077      122      115      126  CNSLO3: .ASCIZ  @RMVEC=@
16 064106      040      114      111  CNSLO4: .ASCIZ  @ LIMITS - LO= 0, HI= 1000@<CRLF><LF>
17 064142      200      124      131  CNSLO7: .ASCII  <CRLF>@TYPE 'A' TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S)@
18 064227      200      101      116      .ASCIZ  <CRLF>@AND TERMINATE INPUT WITH A CARRIAGE RETURN.@
19 064304      200      CNSLO8: .ASCII  <CRLF>
20 064305      040      077      111  CNSLO9: .ASCIZ  @ ?ILLEGAL INPUT@<CRLF>
21 064326      200      104      122  MSDRVS: .ASCIZ  <CRLF>/DRIVE(S): /
22 064342      104      122      111  MSGDRV: .ASCIZ  /DRIVE/
23 064350      200      125      116  SYSTAT: .ASCIZ  <CRLF>/UNIT STATUS:/
24 064366      122      115      060  $RM02: .ASCIZ  /RM02/
25 064373      122      115      060  $RM03: .ASCIZ  /RM03/
26 064400      122      115      060  $RM05: .ASCIZ  /RM05/
27 064405      040      116      117  NOTRM: .ASCIZ  @ NOT AN RM05/3/2@
28 064426      040      114      117  LODEV: .ASCIZ  / LOAD DEVICE/
29 064443      040      116      117  NOTPRS: .ASCIZ  / NOT PRESENT/
30 064460      040      116      117  NOTAVL: .ASCIZ  / NOT AVAILABLE/
31 064477      040      117      106  UNTOFF: .ASCIZ  / OFFLINE/
32 064510      040      117      116  UNTON: .ASCIZ  / ONLINE/
33 064520      200      104      122  DRIVES: .ASCIZ  <CRLF>/DRIVE(S) TO BE TESTED/
34 064547      116      117      116  NONE: .ASCIZ  /NONE/
35 064554      116      000      N: .ASCIZ  /N/
36 064556      131      000      Y: .ASCIZ  /Y/
37 064560      040      BLNKS4: .ASCII  / /
38 064561      040      BLNKS3: .ASCII  / /
39 064562      040      BLNKS2: .ASCII  / /
40 064563      040      000      BLNKS1: .ASCIZ  / /
41      .EVEN
    
```

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

064566
064566 020000

```
.SBTTL FUNCTION CODE TABLE

:THE FUNCTION CODE TABLE IS USED TO DEFINE STATUS CONDITIONS FOR
:EACH FUNCTION CODE. BIT USAGE IS AS FOLLOWS:

:   ATA - BIT 15 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF ATA SHOULD BE SET WHEN TH FUNCTION CODE IS EXECUTED, OTHERWISE,
:   BIT 15 IS ZERO, INDICATING THAT ATA SHOULD NOT NORMALLY BE SET.
:   NOTE THAT ATA MAY BE SET WHEN A COMMAND IS EXECUTED EVEN THOUGH
:   IT IS NOT EXPECTED AS A RESULT OF THE COMMAND.

:   WCE - BIT 14 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF WRITE CHECK ERRORS ARE ENABLED AS A FUNCTION OF THE COMMAND.

:   OPI - BIT 13 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF OPI ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

:   IVC - BIT 12 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF IVC ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

:   WLE - BIT 11 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
:   THE WRITE ERRORS WHICH ARE ENABLED ARE 'WLE', 'WCF', 'DPE', 'UPE'.

:   IAE - BIT 10 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF INVALID ADDRESS ERROR IS ENABLED FOR THAT COMMAND.

:   AOE - BIT 09 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF READ AND WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THE
:   COMMAND. THE ERRORS ENABLED BY THIS BIT ARE 'TRE', 'DLT', 'NEM',
:   'MXF', 'LBT', AND 'AOE'.

:   BIT 08 IS NOT USED.

:   HCE - BIT 07 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF HEADER ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
:   HEADER ERRORS INCLUDE 'HCRC', 'HCE', 'FER', AND 'BSE'.

:   ECH - BIT 06 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
:   IF DATA FIELD ERRORS ARE ENABLED DURING THE EXECUTION OF THAT
:   COMMAND. THESE ERRORS INCLUDE 'MDPE', 'DCK', AND 'ECH'.

:   BIT 05 IS NOT USED.

:   BIT 04 IS NOT USED.

:   BIT 03 IS NOT USED.

:   BIT 02 IS NOT USED.

:   BIT 01 IS NOT USED.

:   ILF - BIT 00 IS SET IF THE FUNCTION CODE IS ILLEGAL.

FNCDTB: ;FUNCTION CODE TABLE

.WORD OPI ;NOP
```

58	064570	130001	.WORD	OPI.ATA.ILF.IVC	:ILLEGAL FUNCTION (2)
59	064572	132000	.WORD	ATA.OPI.IVC IAE	:SEEK
60	064574	130000	.WORD	ATA.OPI.IVC	:RECALIBRATE
61	064576	020000	.WORD	OPI	:DRIVE CLEAR
62	064600	030000	.WORD	OPI!IVC	:RELEASE
63	064602	130000	.WORD	OPI!ATA!IVC	:OFFSET
64	064604	130000	.WORD	OPI!ATA.IVC	:RETURN TO CENTERLINE
65	064606	020000	.WORD	OPI	:READ IN PRESET
66	064610	020000	.WORD	OPI	:PACK ACKNOWLEDGE
67	064612	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (24)
68	064614	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (26)
69	064616	132000	.WORD	ATA!OPI!IVC!IAE	:SEARCH
70	064620	130001	.WORD	OPI.ATA!ILF!IVC	:ILLEGAL FUNCTION (32)
71	064622	130001	.WORD	OPI!ATA.ILF.IVC	:ILLEGAL FUNCTION (34)
72	064624	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (36)
73	064626	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (40)
74	064630	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (42)
75	064632	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (44)
76	064634	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (46)
77	064636	073300	.WORD	WCE!OPI!IVC!IAE!AOE!HCE!ECH	:WRITE CHECK DATA
78	064640	073300	.WORD	WCE!OPI!IVC!IAE!AOE!HCE!ECH	:WRITE CHECK HEADER AND DATA
79	064642	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (54)
80	064644	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (56)
81	064646	037200	.WORD	OPI!IVC!WLE!IAE!AOE!HCE	:WRITE DATA
82	064650	037000	.WORD	OPI!IVC!WLE!IAE!AOE	:WRITE HEADER AND DATA
83	064652	130001	.WORD	OPI.ATA!ILF!IVC	:ILLEGAL FUNCTION (64)
84	064654	130001	.WORD	OPI.ATA!ILF!IVC	:ILLEGAL FUNCTION (66)
85	064656	033300	.WORD	OPI.IVC!IAE!AOE!HCE.ECH	:READ DATA
86	064660	033300	.WORD	OPI!IVC!IAE!AOE!HCE.ECH	:READ HEADER AND DATA
87	064662	130001	.WORD	OPI.ATA!ILF.IVC	:ILLEGAL FUNCTION (74)
88	064664	130001	.WORD	OPI!ATA.ILF!IVC	:ILLEGAL FUNCTION (76)

1		
2		
3	064666	001
4	064667	002
5	064670	004
6	064671	010
7	064672	020
8	064673	040
9	064674	100
0	064675	200

.SBTTL ATTENTION (ATA) TABLE

ATNTBL:	.BYTE	1.
	.BYTE	2.
	.BYTE	4.
	.BYTE	8.
	.BYTE	16.
	.BYTE	32.
	.BYTE	64.
	.BYTE	123.

1
 2
 3 064676
 4 064676
 5 064676 000000
 6 064700 000001
 7 064702 000003
 8 064704 000007
 9 064706 000017
 10 064710 000037
 11 064712 000077
 12 064714 000177
 13 064716 000377
 14 064720 000777
 15 064722 001777
 16 064724 003777
 17 064726 007777
 18 064730 017777
 19 064732 037777
 20 064734 077777
 21 064736 177777
 22 064740 177777
 23 064742 077777
 24 064744 037777
 25 064746 017777
 26 064750 007777
 27 064752 003777
 28 064754 001777
 29 064756 000777
 30 064760 000377
 31 064762 000177
 32 064764 000077
 33 064766 000037
 34 064770 000017
 35 064772 000007
 36 064774 000003
 37 064776 000001
 38 065000 000000
 39 065002 000000
 40 065004 000001
 41 065006 000002
 42 065010 000004
 43 065012 000010
 44 065014 000020
 45 065016 000040
 46 065020 000100
 47 065022 000200
 48 065024 000400
 49 065026 001000
 50 065030 002000
 51 065032 004000
 52 065034 010000
 53 065036 020000
 54 065040 040000
 55 065042 100000
 56 065044 100000
 57 065046 040000

.SBI'L DATA PATTERN TABLE
 RGDTP: .WORD 0.
 MIXED: .WORD 1.
 .WORD 3.
 .WORD 7.
 .WORD 15.
 .WORD 31.
 .WORD 63.
 .WORD 127.
 .WORD 255.
 .WORD 511.
 .WORD 1023.
 .WORD 2047.
 .WORD 4095.
 .WORD 8191.
 .WORD 16383.
 ONES: .WORD 32767.
 .WORD 65535.
 .WORD 65535.
 .WORD 32767.
 .WORD 16383.
 .WORD 8191.
 .WORD 4095.
 .WORD 2047.
 .WORD 1023.
 .WORD 511.
 .WORD 255.
 .WORD 127.
 .WORD 63.
 .WORD 31.
 .WORD 15.
 .WORD 7.
 .WORD 3.
 .WORD 1.
 ZEROS: .WORD 0.
 .WORD 0.
 .WORD 1.
 .WORD 2.
 .WORD 4.
 .WORD 8.
 .WORD 16.
 .WORD 32.
 .WORD 64.
 .WORD 128.
 .WORD 256.
 .WORD 512.
 .WORD 1024.
 .WORD 2048.
 .WORD 4096.
 .WORD 8192.
 .WORD 16384.
 .WORD 32768.
 .WORD 32768.
 .WORD 16384.

58	065050	020000	.WORD	8192.
59	065052	010000	.WORD	4096.
60	065054	004000	.WORD	2048.
61	065056	002000	.WORD	1024.
62	065060	001000	.WORD	512.
63	065062	000400	.WORD	256.
64	065064	000200	.WORD	128.
65	065066	000100	.WORD	64.
66	065070	000040	.WORD	32.
67	065072	000020	.WORD	16.
68	065074	000010	.WORD	8.
69	065076	000004	.WORD	4.
70	065100	000002	.WORD	2.
71	065102	000001	.WORD	1.
72	065104	000000	.WORD	0.
73	065106	177777	.WORD	65535.
74	065110	177776	.WORD	65534.
75	065112	177774	.WORD	65532.
76	065114	177770	.WORD	65528.
77	065116	177760	.WORD	65520.
78	065120	177740	.WORD	65504.
79	065122	177700	.WORD	65472.
80	065124	177600	.WORD	65408.
81	065126	177400	.WORD	65280.
82	065130	177000	.WORD	65024.
83	065132	176000	.WORD	64512.
84	065134	174000	.WORD	63488.
85	065136	170000	.WORD	61440.
86	065140	160000	.WORD	57344.
87	065142	140000	.WORD	49152.
88	065144	100000	.WORD	32768.
89	065146	000000	.WORD	0.
90	065150	000000	.WORD	0.
91	065152	100000	.WORD	32768.
92	065154	140000	.WORD	49152.
93	065156	160000	.WORD	57344.
94	065160	170000	.WORD	61440.
95	065162	174000	.WORD	63488.
96	065164	176000	.WORD	64512.
97	065166	177000	.WORD	65024.
98	065170	177400	.WORD	65280.
99	065172	177600	.WORD	65408.
100	065174	177700	.WORD	65472.
101	065176	177740	.WORD	65504.
102	065200	177760	.WORD	65520.
103	065202	177770	.WORD	65528.
104	065204	177774	.WORD	65532.
105	065206	177776	.WORD	65534.
106	065210	177777	.WORD	65535.
107	065212	125252	.WORD	43690.
108	065214	152525	.WORD	43690. /2
109	065216	125252	.WORD	43690.
110	065220	177777	.WORD	65535.
111	065222	177776	.WORD	65534.
112	065224	177775	.WORD	65533.
113	065226	177773	.WORD	65531.
114	065230	177767	.WORD	65527.

EARLY:

115	065232	177757	.WORD	65519.
116	065234	177737	.WORD	65503.
117	065236	177677	.WORD	65471.
118	065240	177577	.WORD	65407.
119	065242	177377	.WORD	65279.
120	065244	176777	.WORD	65023.
121	065246	175777	.WORD	64511.
122	065250	173777	.WORD	63487.
123	065252	167777	.WORD	61439.
124	065254	157777	.WORD	57343.
125	065256	137777	.WORD	49151.
126	065260	077777	.WORD	32767.
127	065262	077777	.WORD	32767.
128	065264	137777	.WORD	49151.
129	065266	157777	.WORD	57343.
130	065270	167777	.WORD	61439.
131	065272	173777	.WORD	63487.
132	065274	175777	.WORD	64511.
133	065276	176777	.WORD	65023.
134	065300	177377	.WORD	65279.
135	065302	177577	.WORD	65407.
136	065304	177677	.WORD	65471.
137	065306	177737	.WORD	65503.
138	065310	177757	.WORD	65519.
139	065312	177767	.WORD	65527.
140	065314	177773	.WORD	65531.
141	065316	177775	.WORD	65533.
142	065320	177776	.WORD	65534.
143	065322	177777	.WORD	65535.
144	065324			

ENRGDT:

.SBTTL ERROR MESSAGE TABLE

3	065324	071712	000000		EMT1:	.WORD	EMS1,0
4	065330	071761	071776	000000	EMT2:	.WORD	EMS2,EMS3,0
5	065336	071761	072041	000000	EMT3:	.WORD	EMS2,EMS4,0
6	065344	072104	072134	000000	EMT4:	.WORD	EMS5,EMS6,0
7	065352	072104	072246	000000	EMT5:	.WORD	EMS5,EMS10,0
8	065360	076707	074125	000000	EMT6:	.WORD	EMS167,EMS64,0
9	065366	074673	076734	000000	EMT7:	.WORD	EMS110,EMS170,0
10	065374	072201	000000		EMT10:	.WORD	EMS7,0
11	065400	072246	000000		EMT11:	.WORD	EMS10,0
12	065404	072310	072321	000000	EMT12:	.WORD	EMS11,EMS12,0
13	065412	072362	072373	072404	EMT13:	.WORD	EMS13,EMS14,EMS15,EMS16,0
14	065424	072456	074125	000000	EMT14:	.WORD	EMS17,EMS64,0
15	065432	072310	072541	000000	EMT15:	.WORD	EMS11,EMS21,0
16	065440	072310	072564	072713	EMT16:	.WORD	EMS11,EMS22,EMS27,0
17	065450	072310	072600	072724	EMT17:	.WORD	EMS11,EMS23,EMS30,0
18	065460	072310	072626	072724	EMT20:	.WORD	EMS11,EMS24,EMS30,0
19	065470	072310	072655	072713	EMT21:	.WORD	EMS11,EMS25,EMS27,0
20	065500	072310	072672	072713	EMT22:	.WORD	EMS11,EMS26,EMS27,0
21	065510	072310	072734	072724	EMT23:	.WORD	EMS11,EMS31,EMS30,0
22	065520	072310	072763	072724	EMT24:	.WORD	EMS11,EMS32,EMS30,0
23	065530	072310	073012	072724	EMT25:	.WORD	EMS11,EMS33,EMS30,0
24	065540	072310	073040	072724	EMT26:	.WORD	EMS11,EMS34,EMS30,0
25	065550	072310	073111	072724	EMT27:	.WORD	EMS11,EMS35,EMS30,0
26	065560	072310	073140	072724	EMT30:	.WORD	EMS11,EMS36,EMS30,0
27	065570	072310	073167	072724	EMT31:	.WORD	EMS11,EMS37,EMS30,0
28	065600	072310	073215	072724	EMT32:	.WORD	EMS11,EMS40,EMS30,0
29	065610	072310	073244	072724	EMT33:	.WORD	EMS11,EMS41,EMS30,0
30	065620	072310	073272	072724	EMT34:	.WORD	EMS11,EMS42,EMS30,0
31	065630	072310	073321	072724	EMT35:	.WORD	EMS11,EMS43,EMS30,0
32	065640	072310	073350	072724	EMT36:	.WORD	EMS11,EMS44,EMS30,0
33	065650	072310	073423	072724	EMT37:	.WORD	EMS11,EMS45,EMS30,0
34	065660	074211	072516	000000	EMT40:	.WORD	EMS66,EMS20,0
35	065666	074377	076110	074405	EMT41:	.WORD	EMS75,EMS141,EMS76,0
36	065676	076201	076211	074333	EMT42:	.WORD	EMS144,EMS145,EMS72,EMS76,0
37	065710	073515	073631	074405	EMT43:	.WORD	EMS47,EMS53,EMS76,0
38	065720	074436	073631	074405	EMT44:	.WORD	EMS77,EMS53,EMS76,0
39	065730	074464	073631	074405	EMT45:	.WORD	EMS100,EMS53,EMS76,0
40	065740	074512	073631	074405	EMT46:	.WORD	EMS101,EMS53,EMS76,0
41	065750	074143	074125	000000	EMT47:	.WORD	EMS65,EMS64,0
42	065756	072362	072404	074077	EMT50:	.WORD	EMS13,EMS15,EMS63,0
43	065766	072310	073466	072724	EMT51:	.WORD	EMS11,EMS46,EMS30,EMS67,0
44	066000	073515	073631	074261	EMT52:	.WORD	EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
45	066016	073515	073631	074261	EMT53:	.WORD	EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
46	066034	073544	073631	074261	EMT54:	.WORD	EMS50,EMS53,EMS67,0
47	066044	076136	076153	073631	EMT55:	.WORD	EMS142,EMS143,EMS53,EMS67,0
48	066056	073573	074333	074261	EMT56:	.WORD	EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
49	066074	076707	074343	074261	EMT57:	.WORD	EMS167,EMS73,EMS67,0
50	066104	073714	074261	075073	EMT60:	.WORD	EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
51	066122	074320	073714	074261	EMT61:	.WORD	EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
52	066142	076067	074261	075073	EMT62:	.WORD	EMS140,EMS67,EMS115,EMS47,EMS70,0
53	066156	000000			EMT63:	.WORD	
54	066160	076274	076337	074306	EMT64:	.WORD	EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
55	066200	073621	077334	077515	EMT65:	.WORD	EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
56	066220	076646	074567	074333	EMT66:	.WORD	EMS165,EMS103,EMS72,EMS124,0
57	066232	076646	074567	074333	EMT67:	.WORD	EMS165,EMS103,EMS72,EMS171,0

58	066244	073466	072516	076541	EMT70:	.WORD	EMS46,EMS20,EMS163,0
59	066254	074320	074512	076541	EMT71:	.WORD	EMS71,EMS101,EMS163,0
60	066264	073515	074333	076541	EMT72:	.WORD	EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
61	066302	073515	074333	076541	EMT73:	.WORD	EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
62	066320	073714	073631	076541	EMT74:	.WORD	EMS56,EMS53,EMS163,0
63	066330	074320	073714	076541	EMT75:	.WORD	EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
64	066350	073544	073631	076541	EMT76:	.WORD	EMS50,EMS53,EMS163,0
65	066360	076136	076153	073631	EMT77:	.WORD	EMS142,EMS143,EMS53,EMS163,0
66	066372	074377	076110	076541	EMT100:	.WORD	EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
67	066410	074377	076274	076541	EMT101:	.WORD	EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
68	066426	076707	074343	076541	EMT102:	.WORD	EMS167,EMS73,EMS163,0
69	066436	076257	076313	074360	EMT103:	.WORD	EMS147,EMS151,EMS74,EMS163,0
70	066450	073573	074360	076541	EMT104:	.WORD	EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
71	066466	076646	074464	076541	EMT105:	.WORD	EMS165,EMS100,EMS163,0
72	066476	076646	074436	076541	EMT106:	.WORD	EMS165,EMS77,EMS163,0
73	066506	076201	076211	073631	EMT107:	.WORD	EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
74	066526	074673	074733	075076	EMT110:	.WORD	EMS110,EMS112,EMS116,EMS111,0
75	066540	075017	072041	000000	EMT111:	.WORD	EMS113,EMS4,0
76	066546	075017	071776	000000	EMT112:	.WORD	EMS113,EMS3,0
77	066554	074673	075073	075076	EMT113:	.WORD	EMS110,EMS115,EMS116,EMS117,EMS114,0
78	066570	075017	075150	000000	EMT114:	.WORD	EMS113,EMS120,0
79	066576	075171	075631	075655	EMT115:	.WORD	EMS121,EMS132,EMS133,0
80	066606	075234	075631	075655	EMT116:	.WORD	EMS122,EMS132,EMS133,0
81	066616	075271	075631	075655	EMT117:	.WORD	EMS123,EMS132,EMS133,0
82	066626	075334	075631	075655	EMT120:	.WORD	EMS124,EMS132,EMS133,0
83	066636	075366	075631	075655	EMT121:	.WORD	EMS125,EMS132,EMS133,0
84	066646	075431	075631	075655	EMT122:	.WORD	EMS126,EMS132,EMS133,0
85	066656	076032	075631	075655	EMT123:	.WORD	EMS137,EMS132,EMS133,0
86	066666	075533	075631	075655	EMT124:	.WORD	EMS130,EMS132,EMS133,0
87	066676	075571	075631	075655	EMT125:	.WORD	EMS131,EMS132,EMS133,0
88	066706	075171	075631	075700	EMT126:	.WORD	EMS121,EMS132,EMS134,EMS123,0
89	066720	075234	075631	075700	EMT127:	.WORD	EMS122,EMS132,EMS134,EMS123,0
90	066732	075271	075631	075700	EMT130:	.WORD	EMS123,EMS132,EMS134,EMS123,0
91	066744	075334	075631	075700	EMT131:	.WORD	EMS124,EMS132,EMS134,EMS123,0
92	066756	075366	075631	075700	EMT132:	.WORD	EMS125,EMS132,EMS134,EMS123,0
93	066770	075431	075631	075700	EMT133:	.WORD	EMS126,EMS132,EMS134,EMS123,0
94	067002	076032	075631	075700	EMT134:	.WORD	EMS137,EMS132,EMS134,EMS123,0
95	067014	075533	075631	075700	EMT135:	.WORD	EMS130,EMS132,EMS134,EMS123,0
96	067026	075571	075631	075700	EMT136:	.WORD	EMS131,EMS132,EMS134,EMS123,0
97	067040	075171	075631	075742	EMT137:	.WORD	EMS121,EMS132,EMS135,0
98	067050	075271	075631	075742	EMT140:	.WORD	EMS123,EMS132,EMS135,0
99	067060	075171	075631	076005	EMT141:	.WORD	EMS121,EMS132,EMS136,0
100	067070	076032	075631	076005	EMT142:	.WORD	EMS137,EMS132,EMS136,0
101	067100	075334	075631	076005	EMT143:	.WORD	EMS124,EMS132,EMS136,0
102	067110	075366	075631	076005	EMT144:	.WORD	EMS125,EMS132,EMS136,0
103	067120	075431	075631	076005	EMT145:	.WORD	EMS126,EMS132,EMS136,0
104	067130	075571	075631	076005	EMT146:	.WORD	EMS131,EMS132,EMS136,0
105	067140	076763	075631	076005	EMT147:	.WORD	EMS171,EMS132,EMS136,0
106	067150	075533	075631	076005	EMT150:	.WORD	EMS130,EMS132,EMS136,0
107	067160	076067	075073	076110	EMT151:	.WORD	EMS140,EMS115,EMS141,EMS70,0
108	067172	076136	075073	076153	EMT152:	.WORD	EMS142,EMS115,EMS143,EMS72,0
109	067204	076201	076211	076240	EMT153:	.WORD	EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
110	067222	076201	076211	072516	EMT154:	.WORD	EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
111	067240	076274	076337	076405	EMT155:	.WORD	EMS150,EMS152,EMS154,EMS153,0
112	067252	076257	076313	076405	EMT156:	.WORD	EMS147,EMS151,EMS154,EMS155,0
113	067264	076257	076313	076441	EMT157:	.WORD	EMS147,EMS151,EMS156,EMS157,0
114	067276	076511	076441	076474	EMT160:	.WORD	EMS161,EMS156,EMS160,0

115	067306	072655	072713	076441	EMT161: .WORD	EMS25,EMS27,EMS156,EMS160,0
116	067320	072672	072713	076441	EMT162: .WORD	EMS26,EMS27,EMS156,EMS160,0
117	067332	073515	076541	076067	EMT163: .WORD	EMS47,EMS163,EMS140,0
118	067342	073515	072516	000000	EMT164: .WORD	EMS47,EMS20,0
119	067350	073714	072516	000000	EMT165: .WORD	EMS56,EMS20,0
120	067356	073466	072516	000000	EMT166: .WORD	EMS46,EMS20,0
121	067364	100035	072516	000000	EMT167: .WORD	EMS224,EMS20,0
122	067372	077273	076110	072724	EMT170: .WORD	EMS203,EMS141,EMS30,EMS202,0
123	067404	076707	076405	076457	EMT171: .WORD	EMS167,EMS154,EMS157,0
124	067414	076707	076405	076421	EMT172: .WORD	EMS167,EMS154,EMS155,0
125	067424	076763	075631	075655	EMT173: .WORD	EMS171,EMS132,EMS133,0
126	067434	076763	075631	075700	EMT174: .WORD	EMS171,EMS132,EMS134,EMS123,0
127	067446	075017	077136	000000	EMT175: .WORD	EMS113,EMS177,0
128	067454	077160	077175	000000	EMT176: .WORD	EMS200,EMS201,0
129	067462	000000			EMT177: .WORD	
130	067464	077273	073573	072724	EMT200: .WORD	EMS203,EMS51,EMS30,EMS202,0
131	067476	077273	077306	072724	EMT201: .WORD	EMS203,EMS204,EMS30,EMS202,0
132	067510	077273	073544	072724	EMT202: .WORD	EMS203,EMS50,EMS30,EMS202,0
133	067522	077273	076153	072724	EMT203: .WORD	EMS203,EMS143,EMS30,EMS202,0
134	067534	076274	074360	077243	EMT204: .WORD	EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
135	067552	073544	074567	074333	EMT205: .WORD	EMS50,EMS103,EMS72,0
136	067562	074576	074343	077243	EMT206: .WORD	EMS104,EMS73,EMS202,0
137	067572	074377	076110	075073	EMT207: .WORD	EMS75,EMS141,EMS115,EMS140,0
138	067604	074377	076274	000000	EMT210: .WORD	EMS75,EMS150,0
139	067612	073573	074333	075073	EMT211: .WORD	EMS51,EMS72,EMS115,EMS50,EMS70,0
140	067626	076136	073631	075073	EMT212: .WORD	EMS142,EMS53,EMS115,EMS143,EMS72,0
141	067642	073544	074567	073631	E' 213: .WORD	EMS50,EMS103,EMS53,0
142	067652	073621	077334	076662	E' 214: .WORD	EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
143	067672	073621	075126	073714	EMT215: .WORD	EMS52,EMS117,EMS56,EMS163,0
144	067704	073714	072724	074125	EMT216: .WORD	EMS56,EMS30,EMS64,0
145	067714	073466	072724	074125	EMT217: .WORD	EMS46,EMS30,EMS64,0
146	067724	072734	072724	074125	EMT220: .WORD	EMS31,EMS30,EMS64,0
147	067734	073515	072724	074125	EMT221: .WORD	EMS47,EMS30,EMS64,0
148	067744	073621	075126	074436	EMT222: .WORD	EMS52,EMS117,EMS77,0
149	067754	073621	075126	074077	EMT223: .WORD	EMS52,EMS117,EMS63,0
150	067764	000000			EMT224: .WORD	
151	067766	000000			EMT225: .WORD	
152	067770	000000			EMT226: .WORD	
153	067772	000000			EMT227: .WORD	
154	067774	000000			EMT230: .WORD	
155	067776	000000			EMT231: .WORD	
156	070000	000000			EMT232: .WORD	
157	070002	000000			EMT233: .WORD	
158	070004	000000			EMT234: .WORD	
159	070006	000000			EMT235: .WORD	
160	070010	000000			EMT236: .WORD	
161	070012	000000			EMT237: .WORD	
162	070014	000000			EMT240: .WORD	
163	070016	000000			EMT241: .WORD	
164	070020	000000			EMT242: .WORD	
165	070022	000000			EMT243: .WORD	
166	070024	000000			EMT244: .WORD	
167	070026	000000			EMT245: .WORD	
168	070030	076707	075631	077373	EMT246: .WORD	EMS167,EMS132,EMS207,0
169	070040	076707	075631	077420	EMT247: .WORD	EMS167,EMS132,EMS210,EMS125,0
170	070052	076707	077431	077420	EMT250: .WORD	EMS167,EMS211,EMS210,EMS207,EMS206,0
171	070066	077447	077472	000000	EMT251: .WORD	EMS212,EMS213,0

172	070074	076646	077447	000000	EMT252: .WORD	EMS165,EMS212,0
173	070102	076257	076313	076441	EMT253: .WORD	EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
174	070120	077273	074512	072724	EMT254: .WORD	EMS203,EMS101,EMS30,0
175	070130	077273	076707	072724	EMT255: .WORD	EMS203,EMS167,EMS30,0
176	070140	077273	074464	072724	EMT256: .WORD	EMS203,EMS100,EMS30,0
177	070150	072310	073466	072724	EMT257: .WORD	EMS11,EMS46,EMS30,EMS102,0
178	070162	073621	075126	073714	EMT260: .WORD	EMS52,EMS117,EMS56,EMS102,0
179	070174	073621	077334	077637	EMT261: .WORD	EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
180	070214	074576	074343	077243	EMT262: .WORD	EMS104,EMS73,EMS202,0
181	070224	073544	073631	074540	EMT263: .WORD	EMS50,EMS53,EMS102,0
182	070234	073714	073631	074540	EMT264: .WORD	EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
183	070254	074320	073714	074540	EMT265: .WORD	EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
184	070274	076136	076153	073631	EMT266: .WORD	EMS142,EMS143,EMS53,EMS102,0
185	070306	073544	076240	075073	EMT267: .WORD	EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
186	070324	073515	073631	074540	EMT270: .WORD	EMS47,EMS53,EMS102,EMS115,EMS140,0
187	070340	073515	073631	074540	EMT27: .WORD	EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
188	070356	074377	076110	074540	EMT272: .WORD	EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
189	070374	073573	074360	074540	EMT273: .WORD	EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
190	070412	074077	073631	073771	EMT274: .WORD	EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
191	070430	075076	073631	073244	EMT275: .WORD	EMS116,EMS53,EMS41,EMS57,0
192	070442	073515	073631	073771	EMT276: .WORD	EMS47,EMS53,EMS57,EMS115,EMS140,0
193	070456	073515	073631	073771	EMT277: .WORD	EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
194	070474	073714	073631	073771	EMT300: .WORD	EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
195	070514	074320	073714	073631	EMT301: .WORD	EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
196	070536	076646	074464	074567	EMT302: .WORD	EMS165,EMS100,EMS103,EMS57,0
197	070550	076646	074512	074567	EMT303: .WORD	EMS165,EMS101,EMS103,EMS57,0
198	070562	076646	074436	074567	EMT304: .WORD	EMS165,EMS77,EMS103,EMS57,0
199	070574	073466	072724	074125	EMT305: .WORD	EMS46,EMS30,EMS64,EMS57,0
200	070606	073544	073631	073771	EMT306: .WORD	EMS50,EMS53,EMS57,0
201	070616	073544	076240	075073	EMT307: .WORD	EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
202	070636	076136	076153	073631	EMT310: .WORD	EMS142,EMS143,EMS53,EMS57,0
203	070650	074576	074567	073631	EMT311: .WORD	EMS104,EMS103,EMS53,EMS57,0
204	070662	074625	074567	073631	EMT312: .WORD	EMS105,EMS103,EMS53,EMS57,0
205	070674	076201	076211	074567	EMT313: .WORD	EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
206	070714	073272	074567	073631	EMT314: .WORD	EMS42,EMS103,EMS53,EMS57,0
207	070726	072734	074567	073631	EMT315: .WORD	EMS31,EMS103,EMS53,EMS57,0
208	070740	074320	072734	074567	EMT316: .WORD	EMS71,EMS31,EMS103,EMS57,0
209	070752	073321	074567	073771	EMT317: .WORD	EMS43,EMS103,EMS57,0
210	070762	073423	074567	073771	EMT320: .WORD	EMS45,EMS103,EMS57,0
211	070772	073350	074567	073771	EMT321: .WORD	EMS44,EMS103,EMS57,0
212	071002	074654	072516	000000	EMT322: .WORD	EMS106,EMS20,0
213	071010	073140	074567	073771	EMT323: .WORD	EMS36,EMS103,EMS57,0
214	071020	077036	073140	074567	EMT324: .WORD	EMS173,EMS36,EMS103,EMS57,0
215	071032	077016	073140	074567	EMT325: .WORD	EMS172,EMS36,EMS103,EMS57,0
216	071044	072362	077053	072404	EMT326: .WORD	EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
217	071062	076257	076313	074360	EMT327: .WORD	EMS147,EMS151,EMS74,EMS175,0
218	071074	074211	073631	077061	EMT330: .WORD	EMS66,EMS53,EMS175,0
219	071104	073012	074567	073631	EMT331: .WORD	EMS33,EMS103,EMS53,EMS175,0
220	071116	073215	074567	073631	EMT332: .WORD	EMS40,EMS103,EMS53,EMS57,0
221	071130	073573	074360	073771	EMT333: .WORD	EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
222	071146	074377	076110	073771	EMT334: .WORD	EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
223	071164	074377	076274	076337	EMT335: .WORD	EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
224	071204	074017	074032	074061	EMT336: .WORD	EMS60,EMS61,EMS62,0
225	071214	075076	075126	073040	EMT337: .WORD	EMS116,EMS117,EMS34,0
226	071224	073040	073631	073643	EMT340: .WORD	EMS34,EMS53,EMS54,EMS111,0
227	071236	073665	073040	000000	EMT341: .WORD	EMS55,EMS34,0
228	071244	073621	075126	073714	EMT342: .WORD	EMS52,EMS117,EMS56,EMS57,0

229	071256	073621	075126	073423	EMT343: .WORD	EMS52,EMS117,EMS45,EMS57,0
230	071270	073621	075126	073350	EMT344: .WORD	EMS52,EMS117,EMS44,EMS57,0
231	071302	073621	075126	077657	EMT345: .WORD	EMS52,EMS117,EMS221,0
232	071312	074654	072516	075073	EMT346: .WORD	EMS106,FMS20,EMS115,EMS223,EMS72,0
233	071326	073621	077334	077727	EMT347: .WORD	EMS52,EMS205,EMS222,EMS206,0
234	071340	074377	076274	074540	EMT350: .WORD	EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
235	071356	076707	074343	074540	EMT351: .WORD	EMS167,EMS73,EMS102,0
236	071366	077533	000000		EMT352: .WORD	EMS215,0
237	071372	077604	077273	077554	EMT353: .WORD	EMS217,EMS203,EMS216,0
238	071402	077657	072516	000000	EMT354: .WORD	EMS221,EMS20,0

1	071410	100107	100713	100770	EHT1:	.WORD	EH1,STSH1,STSH2,STSH4,0
2	071422	100713	100770	101115	EHT2:	.WORD	STSH1,STSH2,STSH4,0
3							
4	071432	100126	000000		EHT110:	.WORD	EH110,0
5	071436	100135	000000		EHT111:	.WORD	EH111,0
6							
7	071442	100154	000000		EHT114:	.WORD	EH114,0
8	071446	100203	100713	100770	EHT223:	.WORD	EH223,STSH1,STSH2,STSH4,0
9	071460	100231	100713	100770	EHT256:	.WORD	EH256,STSH1,STSH2,STSH4,0
10							
11	071472	100305	100713	100770	EHT336:	.WORD	EH336,STSH1,STSH2,STSH4,0
12	071504	100344	100713	100770	EHT337:	.WORD	EH337,STSH1,STSH2,STSH4,0
13	071516	100501	100713	100770	EHT344:	.WORD	EH344,STSH1,STSH2,STSH4,0
14							
15	071530	100637	000000		EHT353:	.WORD	EH353,0

1	071534	101154	101250	101266	EDT1:	.WORD	ED1,STSD1,STSD2,STSD4
2	071544	101250	101266	101320	EDT2:	.WORD	STSD1,STSD2,STSD4
3							
4	071552	101162			EDT110:	.WORD	ED110
5	071554	101166			EDT111:	.WORD	ED111
6							
7	071556	101174			EDT114:	.WORD	ED114
8	071560	101204	101250	101266	EDT223:	.WORD	ED223,STSD1,STSD2,STSD4
9							
10	071570	101214	101250	101266	EDT336:	.WORD	ED336,STSD1,STSD2,STSD4
11	071600	101226	101250	101266	EDT337:	.WORD	ED337,STSD1,STSD2,STSD4
12	071610	101226	101250	101266	EDT344:	.WORD	ED337,STSD1,STSD2,STSD4,0
13							
14	071622	101240			EDT353:	.WORD	ED353

1	071624	101333	101351	101351	EFT1:	.WORD	EF111,STSF,STSF,STSF
2	071634	101351	101351	101351	EFT2:	.WORD	STSF,STSF,STSF
3							
4	071642	101332			EFT110:	.WORD	EF110
5	071644	101333			EFT111:	.WORD	EF111
6							
7	071646	101335			EFT114:	.WORD	EF114
8	071650	101335	101351	101351	EFT223:	.WORD	EF114,STSF,STSF,STSF
9							
10	071660	101340	101351	101351	EFT336:	.WORD	EF336,STSF,STSF,STSF
11	071670	101340	101351	101351	EFT337:	.WORD	EF336,STSF,STSF,STSF
12	071700	101340	101351	101351	EFT344:	.WORD	EF336,STSF,STSF,STSF
13							
14	071710	101335			EFT353:	.WORD	EF114

Line	Hex	Hex	Hex	Label	Description
1				.SBTTL	ERROR MESSAGE STRINGS
2					
3	071712	127	122	117 EMS1:	.ASCIZ @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
4	071761	104	105	126 EMS2:	.ASCIZ @DEVICE WENT @
5	071776	125	116	101 EMS3:	.ASCIZ @UNAVAILABLE 'DVA' (RMCS1, BIT 11) @
6	072041	116	117	116 EMS4:	.ASCIZ @NONEXISTENT 'NED' (RMCS2, BIT 12) @
7	072104	103	117	115 EMS5:	.ASCIZ @COMMAND NOT COMPLETED, @
8	072134	103	117	116 EMS6:	.ASCIZ @CONTROLLER NOT READY (RMCS1, BIT 7) @
9	072201	104	122	111 EMS7:	.ASCIZ @DRIVE NOT READY 'DRY' (RMDS, BIT 7) @
10	072246	107	117	040 EMS10:	.ASCIZ @GO NOT RESET 'GO' (RMCS1, BIT 0) @
11	072310	111	116	126 EMS11:	.ASCIZ @INVALID @
12	072321	106	125	116 EMS12:	.ASCIZ @FUNCTION CODE (RMCS1, BITS 1-5) @
13	072362	115	101	123 EMS13:	.ASCIZ @MASSBUS @
14	072373	103	117	116 EMS14:	.ASCIZ @CONTROL @
15	072404	102	125	123 EMS15:	.ASCIZ @BUS PARITY ERROR @
16	072426	042	115	103 EMS16:	.ASCIZ @'MCPE' (RMCS1, BIT 13) @
17	072456	124	122	101 EMS17:	.ASCIZ @TRANSFER ERROR (RMCS1, BIT 14) @
18	072516	123	110	117 EMS20:	.ASCIZ @SHOULD NOT BE SET @
19	072541	127	117	122 EMS21:	.ASCIZ @WORD COUNT (RMWC) @
20	072564	102	125	123 EMS22:	.ASCIZ @BUS (RMBA) @
21	072600	042	114	102 EMS23:	.ASCIZ @'LBT' (RMDS, BIT 10) @
22	072626	042	101	117 EMS24:	.ASCIZ @'AOE' (RMER1, BIT 09) @
23	072655	104	111	123 EMS25:	.ASCIZ @DISK (RMDA) @
24	072672	103	131	114 EMS26:	.ASCIZ @CYLINDER (RMDC) @
25	072713	101	104	104 EMS27:	.ASCIZ @ADDRESS @
26	072724	123	124	101 EMS30:	.ASCIZ @STATUS @
27	072734	042	127	114 EMS31:	.ASCIZ @'WLE' (RMER1, BIT 11) @
28	072763	042	125	120 EMS32:	.ASCIZ @'UPE' (RMCS2, BIT 13) @
29	073012	042	127	103 EMS33:	.ASCIZ @'WCF' (RMER1, BIT 5) @
30	073040	127	122	111 EMS34:	.ASCIZ @WRITE CHECK ERROR-'WCE' (RMCS2, BIT 14) @
31	073111	042	115	104 EMS35:	.ASCIZ @'MDPE' (RMCS2, BIT 8) @
32	073140	042	104	103 EMS36:	.ASCIZ @'DCK' (RMER1, BIT 15) @
33	073167	042	105	103 EMS37:	.ASCIZ @'ECH' (RMER1, BIT 6) @
34	073215	042	104	114 EMS40:	.ASCIZ @'DLT' (RMCS2, BIT 15) @
35	073244	042	115	130 EMS41:	.ASCIZ @'MXF' (RMCS2, BIT 9) @
36	073272	042	104	124 EMS42:	.ASCIZ @'DTE' (RMER1, BIT 12) @
37	073321	042	110	103 EMS43:	.ASCIZ @'HCRC' (RMER1, BIT 8) @
38	073350	110	105	101 EMS44:	.ASCIZ @HEADER COMPARE ERROR 'HCE' (RMER1, BIT 7) @
39	073423	106	117	122 EMS45:	.ASCIZ @FORMAT ERROR 'FER' (RMER1, BIT 4) @
40	073466	042	111	101 EMS46:	.ASCIZ @'IAE' (RMER1, BIT 10) @
41	073515	042	117	120 EMS47:	.ASCIZ @'DPI' (RMER1, BIT 13) @
42	073544	042	123	113 EMS50:	.ASCIZ @'SKI' (RMER2, BIT 14) @
43	073573	042	120	111 EMS51:	.ASCIZ @'PIP' (RMDS, BIT 13) @
44	073621	124	110	105 EMS52:	.ASCIZ @THE RM @
45	073631	104	105	124 EMS53:	.ASCIZ @DETECTED @
46	073643	101	124	040 EMS54:	.ASCIZ @AT AN UNEXPECTED @
47	073665	111	116	103 EMS55:	.ASCIZ @INCORRECT DATA DURING @
48	073714	111	116	126 EMS56:	.ASCIZ @INVALID COMMAND ERROR 'IVC' (RMER2, BIT 12) @
49	073771	104	125	122 EMS57:	.ASCIZ @DURING DATA TRANSFER @
50	074017	104	101	124 EMS60:	.ASCIZ @DATA READ @
51	074032	104	117	105 EMS61:	.ASCIZ @DOES NOT COMPARE WITH @
52	074061	104	101	124 EMS62:	.ASCIZ @DATA WRITTEN @
53	074077	042	120	101 EMS63:	.ASCIZ @'PAR' (RMER1, BIT 3) @
54	074125	111	123	040 EMS64:	.ASCIZ @IS INCORRECT @
55	074143	103	117	115 EMS65:	.ASCIZ @COMPOSITE ERROR 'ERR' (RMDS, BIT 14) @
56	074211	104	101	124 EMS66:	.ASCIZ @DATA PARITY ERROR 'DPE' (RMER2, BIT 3) @
57	074261	104	125	122 EMS67:	.ASCIZ @DURING SEEK COMMAND @

58	074306	111	123	040	EMS70:	.ASCIZ	@IS RESET @
59	074320	105	122	122	EMS71:	.ASCIZ	@ERRONEOUS @
60	074333	111	123	040	EMS72:	.ASCIZ	@IS SET @
61	074343	104	111	104	EMS73:	.ASCIZ	@DID NOT SET @
62	074360	104	111	104	EMS74:	.ASCIZ	@DID NOT RESET @
63	074377	114	117	123	EMS75:	.ASCIZ	@LOST @
64	074405	104	125	122	EMS76:	.ASCIZ	@DURING PACK ACK COMMAND @
65	074436	042	122	115	EMS77:	.ASCIZ	@'RMR' (RMER1, BIT 2) @
66	074464	042	111	114	EMS100:	.ASCIZ	@'ILR' (RMER1, BIT 1) @
67	074512	042	111	114	EMS101:	.ASCIZ	@'ILF' (RMER1, BIT 0) @
68	074540	104	125	122	EMS102:	.ASCIZ	@DURING SEARCH COMMAND @
69	074567	105	122	122	EMS103:	.ASCIZ	@ERROR @
70	074576	042	114	102	EMS104:	.ASCIZ	@'LBC' (RMER2, BIT 10) @
71	074625	042	114	123	EMS105:	.ASCIZ	@'LSC' (RMER2, BIT 11) @
72	074654	110	105	101	EMS106:	.ASCIZ	@HEADER ERRORS @
73	074673	102	125	123	EMS110:	.ASCIZ	@BUS TIMEOUT (04 TRAP) @
74	074722	101	104	104	EMS111:	.ASCIZ	@ADDRESS @
75	074733	127	110	105	EMS112:	.ASCII	@WHEN READING/WRITING RH REGISTERS @
76	074775	101	124	040		.ASCIZ	@AT THE FOLLOWING @
77	075017	124	110	105	EMS113:	.ASCIZ	@THE SELECTED DEVICE IS @
78	075047	116	117	116	EMS114:	.ASCIZ	@NONEXISTENT DEVICE @
79	075073	015	012	000	EMS115:	.ASCIZ	<CR><LF>
80	075076	124	110	105	EMS116:	.ASCIZ	@THE MASSBUS CONTROLLER @
81	075126	106	101	111	EMS117:	.ASCIZ	@FAILED TO DETECT @
82	075150	116	117	124	EMS120:	.ASCIZ	@NOT AN RM05/3/2 @
83	075171	103	117	116	EMS121:	.ASCIZ	@CONTROL STATUS REGISTER 1, RMCS1, @
84	075234	102	125	123	EMS122:	.ASCIZ	@BUS ADDRESS REGISTER, RMA, @
85	075271	103	117	116	EMS123:	.ASCIZ	@CONTROL STATUS REGISTER 2, RMCS2, @
86	075334	105	122	122	EMS124:	.ASCIZ	@ERROR REGISTER 1, RMER1, @
87	075366	101	124	124	EMS125:	.ASCIZ	@ATTENTION SUMMARY REGISTER, RMAS, @
88	075431	115	101	111	EMS126:	.ASCIZ	@MAINTENANCE REGISTER #1, RMMR #1, @
89	075474	105	103	103	EMS127:	.ASCIZ	@ECC POSITION REGISTER, RMEC1, @
90	075533	105	103	103	EMS130:	.ASCIZ	@ECC PATTERN REGISTER, RMEC2, @
91	075571	115	101	111	EMS131:	.ASCIZ	@MAINTENANCE REGISTER 2, RMMR2, @
92	075631	116	117	124	EMS132:	.ASCIZ	@NOT INITIALIZED BY @
93	075655	125	116	111	EMS133:	.ASCIZ	@UNIBUS INITIALIZE @
94	075700	103	117	116	EMS134:	.ASCIZ	@CONTROLLER CLEAR, I.E. BIT 5 OF @
95	075742	122	110	057	EMS135:	.ASCIZ	@RH/RM ERROR CLEAR (RMCS1, BIT 14) @
96	076005	104	122	111	EMS136:	.ASCIZ	@DRIVE CLEAR COMMAND @
97	076032	104	122	111	EMS137:	.ASCIZ	@DRIVE STATUS REGISTER, RMDS @
98	076067	115	105	104	EMS140:	.ASCIZ	@MEDIUM OFF LINE @
99	076110	042	115	117	EMS141:	.ASCIZ	@'MOL' (RMDS, BIT 12) @
100	076136	104	122	111	EMS142:	.ASCIZ	@DRIVE FAULT @
101	076153	042	104	126	EMS143:	.ASCIZ	@'DVC' (RMER2, BIT 7) @
102	076201	125	116	123	EMS144:	.ASCIZ	@UNSAFE @
103	076211	042	125	116	EMS145:	.ASCIZ	@'UNS' (RMER1, BIT 14) @
104	076240	123	110	117	EMS146:	.ASCIZ	@SHOULD BE SET @
105	076257	117	106	106	EMS147:	.ASCIZ	@OFFSET MODE @
106	076274	040	126	117	FMS150:	.ASCIZ	@ VOLUME VALID @
107	076313	042	117	115	EMS151:	.ASCIZ	@'OM' (RMDS, BIT 0) @
108	076337	042	126	126	EMS152:	.ASCIZ	@'VV' (RMDS, BIT 6) @
109	076363	120	101	103	EMS153:	.ASCIZ	@PACK ACK COMMAND @
110	076405	116	117	124	EMS154:	.ASCIZ	@NOT SET BY @
111	076421	117	106	106	EMS155:	.ASCIZ	@OFFSET COMMAND @
112	076441	116	117	124	EMS156:	.ASCIZ	@NOT RESET BY @
113	076457	122	124	103	EMS157:	.ASCIZ	@RTC COMMAND @
114	076474	122	111	120	EMS150:	.ASCIZ	@RIP COMMAND @

115	076511	117	106	106	EMS161: .ASCIZ @OFFSET REGISTER (RMOF) @
116	076541				EMS162: :<UNUSED>
117	076541	104	125	122	EMS163: .ASCIZ @DURING RECALIBRATE @
118	076565	111	123	040	EMS164: .ASCII @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
119	076634	103	131	114	.ASCIZ @CYLINDER @
120	076646	125	116	105	EMS165: .ASCIZ @UNEXPECTED @
121	076662	122	105	103	EMS166: .ASCIZ @RECALIBRATE COMMAND @
122	076707	042	101	124	EMS167: .ASCIZ @'ATA' (RMDS, BIT15) @
123	076734	127	110	105	EMS170: .ASCIZ @WHEN READING REGISTER @
124	076763	105	122	122	EMS171: .ASCIZ @ERROR REGISTER #2, RMER2, @
125	077016	116	117	116	EMS172: .ASCIZ @NONRECOVERABLE @
126	077036	122	105	103	EMS173: .ASCIZ @RECOVERABLE @
127	077053	104	101	124	EMS174: .ASCIZ @DATA @
128	077061	104	125	122	EMS175: .ASCIZ @DURING WRITE COMMAND @
129	077107	042	117	120	EMS176: .ASCIZ @'OPE' (RMER2, BIT 13) @
130	077136	111	116	040	EMS177: .ASCIZ @IN WRITE PROTECT @
131	077160	103	101	116	EMS200: .ASCIZ @CAN NOT SET @
132	077175	104	111	101	EMS201: .ASCIZ @DIAGNOSTIC MODE 'DMD' (RMMR1, BIT 0) @
133	077243	104	125	122	EMS202: .ASCIZ @DURING DIAGNOSTIC MODE @
134	077273	111	116	103	EMS203: .ASCIZ @INCORRECT @
135	077306	042	127	122	EMS204: .ASCIZ @'WRL' (RMDS, BIT 11) @
136	077334	105	130	105	EMS205: .ASCIZ @EXECUTED @
137	077346	127	111	124	EMS206: .ASCIZ @WITH COMP ERROR SET @
138	077373	042	107	117	EMS207: .ASCIZ @'GO' (RMCS1, BIT 0) @
139	077420	127	122	111	EMS210: .ASCIZ @WRITING @
140	077431	127	101	123	EMS211: .ASCIZ @WAS RESET BY @
141	077447	120	122	117	EMS212: .ASCIZ @PROGRAM INTERRUPT @
142	077472	127	101	123	EMS213: .ASCIZ @WAS NOT GENERATED @
143	077515	123	105	105	EMS214: .ASCIZ @SEEK COMMAND @
144	077533	120	122	117	EMS215: .ASCIZ @PROGRAM TIMEOUT @
145	077554	104	125	122	EMS216: .ASCIZ @DURING LOOK AHEAD TEST @
146	077604	114	117	117	EMS217: .ASCIZ @LOOK AHEAD REGISTER, RMLA, @
147	077637	123	105	101	EMS220: .ASCIZ @SEARCH COMMAND @
148	077657	102	101	104	EMS221: .ASCIZ @BAD SECTOR ERROR 'BSE' (RMER2, BIT 15) @
149	077727	101	040	104	EMS222: .ASCIZ @A DATA TRANSFER COMMAND @
150	077760	110	105	101	EMS223: .ASCIZ @HEADER COMPARE INHIBIT 'HCI' (RMOF, BIT 10) @
151	100035	116	117	116	EMS224: .ASCIZ @NONEXISTENT MEMORY 'NEM' (RMCS2, BIT 11) @

1	100107	105	130	120	EH1:	.ASCIZ	@EXPCTD	RECEVD@			
2	100126	102	125	123	EH110:	.ASCIZ	@BUSADRA				
3	100135	040	122	115	EH111:	.ASCIZ	@ RMCS2	RMCS @			
4											
5	100154	122	105	103	EH114:	.ASCIZ	@RECEVD	SNGPRT	DULPRI@		
6	100203	105	130	120	EH223:	.ASCIZ	@EXPCTD	RECEVD	DATA@		
7	100231	105	130	120	EH256:	.ASCII	@EXPCTD	RECEVD	RGSTR@<CRLF>		
8	100257	123	124	101		.ASCIZ	@STATUS	STATUS	INDEX@		
9											
10	100305	107	104	101	EH336:	.ASCIZ	@GDADRS	GDDATA	BDADRS	BDDATA@	
11	100344	122	115	103	EH337:	.ASCII	@RMCS2	STATUS	FAILING	DATA@<CRLF>	
12	100403	137	137	137		.ASCII	@			@<CRLF>	
13	100442	105	130	120		.ASCIZ	@EXPCTD	RECEVD	--BIT--	ADDRESS@	
14											
15	100501	122	115	105	EH344:	.ASCII	@RMER1	STATUS	HEADER	FAILING@<CRLF>	
16	100541	137	137	137		.ASCII	@		WORD	BIT@<CRLF>	
17	100577	105	130	120		.ASCIZ	@EXPCTD	RECEVD	NUMBER	POSITON@	
18	100637	105	130	120	EH353:	.ASCII	@EXPCTD	RECEVD@<CRLF>			
19	100656	074	103	122		.ASCIZ	@<CRLF>	RMLA	RMLA	RMOF @	
20											
21	100713	040	122	115	STSH1:	.ASCII	@ RMCS1	RMCS2	RMDS	RMER1	RMER2@
22	100760	040	040	040		.ASCIZ	@	RMAS@			
23	100770	040	122	115	STSH2:	.ASCII	@ RMWC	RMBA	RMDA	RMOF	RMDCA
24	101035	040	040	040		.ASCIZ	@	RMEC1	RMEC2@		
25	101057	040	122	115	STSH3:	.ASCIZ	@ RMDA	RMDC	RMOF	RMLA@	
26	101115	040	122	115	STSH4:	.ASCIZ	@ RMMR1	RMMR2	RMDT	RMSN@	
27						.EVEN					

1	101154	001140	001142	000000	ED1:	.WORD	\$GDDAT,\$BDDAT,0
2	101162	001276	000000		ED110:	.WORD	\$BASE,0
3	101166	001174	001176	000000	ED111:	.WORD	\$TMP0,\$TMP1,0
4							
5	101174	001364	001176	001200	ED114:	.WORD	RMDTI,\$TMP1,\$TMP2,0
6	101204	001140	001142	001174	ED223:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,0
7							
8	101214	001134	001140	001136	ED336:	.WORD	\$GDADR,\$GDDAT,\$BDADR,\$BDDAT,0
9							
10	101226	001140	001142	001174	ED337:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,\$TMP1,0
11	101240	001140	001142	001444	ED353:	.WORD	\$GDDAT,\$BDDAT,RMOFO,0
12							
13	101250	001336	001346	001350	STSD1:	.WORD	RMCS1I, RMCS2I, RMDSI, RMER1I, RMER2I, RMASI, 0
14	101266	001340	001342	001344	STSD2:	.WORD	RMWCI, RMBAI, RMDAI, RMOFI, RMDCI, RMECI
15	101302	001404	000000			.WORD	RMEC2I, 0
16	101306	001344	001372	001370	STSD3:	.WORD	RMDAI, RMDCI, RMOFI, RMLAI, 0
17	101320	001362	001376	001364	STSD4:	.WORD	RMMR1I, RMMR2I, RMDTI, RMSNI, 0

1	101332	000			EF110:	.BYTE	0
2	101333	000	000		EF111:	.BYTE	0,0
3	101335	000	000	000	EF114:	.BYTE	0,0,0
4	101340	000	000	000	EF336:	.BYTE	0,0,0,0
5	101344	000	000	000	EF337:	.BYTE	0,0,0,0,0
6							
7	101351	000	000	000	STSF:	.BYTE	0,0,0,0,0,0,0
8					.EVEN		

1
2 101360
3 101360
4 102364
5
6
7 103370 000000 000000
8 103374
9 104374 177777
10
11
12 104376 000000 000000
13 104402
14 105402 177777
15
16 101360
17
18 101360
19 101360 200
20 101361 200
21 101362 114 111 123
22 101400 055 055 055
23 101416 124 061 011
24 101450 124 062 011
25 101470 124 063 011
26 101512 124 064 011
27 101540 124 065 011
28 101603 124 066 011
29 101622 124 067 011
30 101647 124 061 060
31 101713 124 061 061
32 101747 124 061 062
33 102004 124 061 063
34 102044 124 061 064
35 102077 124 061 065
36 102136 124 061 066
37 102174 124 061 067
38 102227 124 062 060
39 102275 124 062 061
40 102341 124 062 062
41 102401 124 062 063
42 102440 124 062 064
43 102502 124 062 065
44 102527 124 062 066
45 102553 124 062 067
46 102601 124 063 060
47 102643 124 063 061
48 102706 200
49 102707 117 120 105
50 102743 055 055 055
51 102777 123 127 111
52 103014 055 055 055
53 103051 040 040 061
54 103075 040 040 061
55 103120 040 040 061
56 103155 040 040 061
57 103164 040 040 061

```
;STORAGE FOR GENERAL DATA TRANSFERS
BUFFER:
BUFONE: .BLKW 258.
BUFTWO: .BLKW 258.

;STORAGE FOR MANUFACTURES 16 BIT MODE BAD SECTOR FILE
MFGFIL: .WORD 0,0 ;2 HEADER WORDS
        .BLKW 256. ;256. WORDS OF DATA
        .WORD -1 ;TERMINATOR IF FILE IS FULL

;STORAGE FOR USERS 16 BIT MODE BAD SECTOR FILE
USRFIL: .WORD 0,0 ;2 HEADER WORDS
        .BLKW 256. ;256. WORDS OF DATA
        .WORD -1 ;TERMINATOR IF FILE IS FULL

.-BUFFER

HELP:
.ASCII <CRLF>
.ASCII <CRLF>
.ASCII @LIST OF TESTS@<CRLF>
.ASCII @-----@<CRLF>
.ASCII @T1 CONTROLLER ACCESS TEST@<CRLF>
.ASCII @T2 FORMAT ZEROS@<CRLF>
.ASCII @T3 ZERO FILL TEST@<CRLF>
.ASCII @T4 FORMAT CHECK ZEROS@<CRLF>
.ASCII @T5 FORMAT CHECK ZEROS W/ WCE ERROR@<CRLF>
.ASCII @T6 FORMAT ONES@<CRLF>
.ASCII @T7 FORMAT CHECK ONES@<CRLF>
.ASCII @T10 FORMAT CHECK ONES W/ WCE ERRORS@<CRLF>
.ASCII @T11 FORMAT MULTIPLE SECTORS@<CRLF>
.ASCII @T12 FORMAT W/ HEAD SWITCHING@<CRLF>
.ASCII @T13 FORMAT W/ MID TRANSFER SEEK@<CRLF>
.ASCII @T14 FORMAT W/ IMPLIED SEEK@<CRLF>
.ASCII @T15 FORMAT EACH SECTOR ADDRESS@<CRLF>
.ASCII @T16 FORMAT EACH TRACK ADDRESS@<CRLF>
.ASCII @T17 FORMAT PRIME CYLINDERS@<CRLF>
.ASCII @T20 READ HEADER & DATA IN LAST SECTOR@<CRLF>
.ASCII @T21 READ HEADER & DATA W/ AOE ERROR@<CRLF>
.ASCII @T22 READ INVALID SECTOR ADDRESS@<CRLF>
.ASCII @T23 READ INVALID TRACK ADDRESS@<CRLF>
.ASCII @T24 READ INVALID CYLINDER ADDRESS@<CRLF>
.ASCII @T25 FORMAT AT OFFSET@<CRLF>
.ASCII @T26 IVC FORMAT TEST@<CRLF>
.ASCII @T27 FORMAT ERROR TEST@<CRLF>
.ASCII @T30 FORMAT HCE, FIRST HEADER WORD@<CRLF>
.ASCII @T31 FORMAT HCE, SECOND HEADER WORD@<CRLF>
.ASCII <CRLF>
.ASCII @OPERATIONAL SWITCH SETTINGS@<CRLF>
.ASCII @-----@<CRLF>
.ASCII @SWITCH USE@<CRLF>
.ASCII @-----@<CRLF>
.ASCII @ 15 HALT ON ERROR@<CRLF>
.ASCII @ 14 LOOP ON TEST@<CRLF>
.ASCII @ 13 INHIBIT ERROR TYPEOUTS@<CRLF>
.ASCII @ 12 @<CRLF>
.ASCII @ 11 INHIBIT ITERATIONS@<CRLF>
```

ERROR MESSAGE STRINGS

58	103215	040	040	061	.ASCII	@	10	BELL ON ERROR@<CRLF>
59	103241	040	040	040	.ASCII	@	9	LOOP ON ERROR@<CRLF>
60	103265	040	040	040	.ASCII	@	8	LOOP ON TEST IN SWR<7:0>@<CRLF>
61	103324	040	040	040	.ASCII	@	7	TN128@<CRLF>
62	103340	040	040	040	.ASCII	@	6	TN64@<CRLF>
63	103353	040	040	040	.ASCII	@	5	TN32@<CRLF>
64	103366	040	040	040	.ASCII	@	4	TN16@<CRLF>
65	103401	040	040	040	.ASCII	@	3	TN8@<CRLF>
66	103413	040	040	040	.ASCII	@	2	TN4@<CRLF>
67	103425	040	040	040	.ASCII	@	1	TN2@<CRLF>
68	103437	040	040	040	.ASCII	@	0	TN1@<CRLF>
69								
70	000200			.END			200	

SYMBOL TABLE

ABASE	176700	ATNMSK-	000377	BUFONE	101360	EBL	=	020000	EH344	100501																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
ACDW1	=	000000	ATNTBL	064666	BUFTWO	102364	ECH	=	000100	EH353	100637																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ACDW2	=	000000	AUNIT	000000	CC	=	004000	ECL	=	004000	EMS1	071712																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
ACKSTS	046342	AUSWR	000000	CH	=	002000	ECRC	=	001000	EMS10	072246																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ACPUOP	=	000000	AVECT1	=	120254	CHGADR	001330	EDT1	071534	EMS100	074464																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ADDW0	=	000000	AVECT2	=	000000	CHRCNT	061027	EDT110	071552	EMS101	074512																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ADDW1	=	000000	A16	=	000400	CKSWR	-	104410	EDT111	071554	EMS102	074540																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
ADDW10	=	000000	A17	=	001000	CLKADR	001520	EDT114	071556	EMS103	074567																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ADDW11	=	000000	BACK	=	000000	CLKVCT	001522	EDT2	071544	EMS104	074576																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ADDW12	=	000000	BADSCT	033152	CLR	=	000040	EDT223	071560	EMS105	074625																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ADDW13	=	000000	BADTMO	005342	CLRSTS	045462	EDT336	071570	EDT337	071600	EMS106	074654																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
ADDW14	=	000000	BAI	=	000010	CMNSTA	007476	EDT344	071610	EDT353	071622	EMS11	072310																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW15	=	000000	BB00	=	000001	CMPBUF	035342	ED1	101154	ED110	101162	EMS110	074673																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW2	=	000000	BB01	=	000002	CMPERR	043460	ED110	101162	ED111	101166	EMS111	074722																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW3	=	000000	BB02	=	000004	CNSL01	064025	ED111	101166	ED114	101174	EMS112	074733																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW4	=	000000	BB03	=	000010	CNSL02	064035	ED223	101204	ED336	101214	EMS113	075017																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW5	=	000000	BB04	=	000020	CNSL03	064077	ED336	101214	ED337	101226	EMS114	075047																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW6	=	000000	BB05	=	000040	CNSL04	064106	ED337	101226	ED353	101240	EMS115	075073																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW7	=	000000	BB06	=	000100	CNSL07	064142	ECC	=	000020	EFT1	071624	EMS116	075076																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
ADDW8	=	000000	BB07	=	000200	CNSL08	064304	EFT110	071642	EFT111	071644	EMS117	075126																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADDW9	=	000000	BB08	=	000400	CNSL09	064305	EFT114	071644	EFT114	071646	EMS12	072321																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADEVCT	=	000000	BB09	=	001000	CNTCLR	045344	EFT2	071634	EFT223	071650	EMS120	075150																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADEVVM	=	000000	BIT0	=	000001	COMMA	063445	EFT336	071660	EFT337	071670	EMS121	075171																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ADR	=	000001	BIT00	=	000001	CONT	=	000100	EFT344	071700	EFT353	071710	EMS122	075234																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
AENV	=	000000	BIT01	=	000002	CPSAVE	060150	CR	=	000015	EF110	101332	EMS123	075271																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
AENVM	000000	AFATAL	=	000000	CR	=	000015	CRLF	=	000200	EF111	101333	EMS124	075334																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
ALL	063434	AMADR1	=	000000	CTLFG	001326	CYLSK	=	001777	EF114	101335	EF336	101340	EMS125	075366																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMADR2	=	000000	BIT02	=	000004	DBCK	100000	DBEN	=	040000	EF337	101344	EF337	101344	EMS126	075431																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
AMADR3	=	000000	BIT03	=	000010	DBL	=	002000	DDISP	=	177570	EHT1	071410	EMS127	075474																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMADR4	=	000000	BIT04	=	000020	DCK	=	100000	DEBL	=	020000	EHT110	071432	EMS13	072362																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMAMS1	=	000000	BIT05	=	000040	DEVSEL	043672	DISPLA	001156	DISPRE	000174	EHT111	071436	EMS130	075533																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMAMS2	=	000000	BIT06	=	000100	DMD	=	000001	DLT	=	100000	EHT114	071442	EMS131	075571																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMAMS3	=	000000	BIT07	=	000200	DPE	=	000010	DPEHI	=	040000	EHT2	071422	EMS132	075631																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMAMS4	=	000000	BIT08	=	000400	DPELO	=	020000	DPR	=	000400	EHT223	071446	EMS133	075655																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
AMSGAD	=	000000	BIT09	=	001000	DRIVES	064520	DRQ	=	004000	EHT256	071460	EHT336	071472	EMS134	075700																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
AMSGLG	=	000000	BIT10	=	002000	DRVCLR	=	000010	DSWR	=	177570	EHT337	071504	EHT344	071516	EMS135	075742																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
AMSGTY	=	000000	BIT11	=	004000	DRVSTS	050700	DTASTS	051502	DTE	=	010000	EH1	100107	EF337	101344	EMS136	076005																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
AMTYP1	=	000000	BIT12	=	010000	DRY	=	000200	DTO	=	010000	EH110	100126	EH111	100135	EH223	100203	EMS137	076032																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
AMTYP2	=	000000	BIT13	=	020000	DVA	=	004000	DULPRT	=	024024	EH114	100154	EH223	100203	EH256	100231	EMS14	072373																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
AMTYP3	=	000000	BIT14	=	040000	DVC	=	000200	EARLY	065212	EH336	100305	EH337	100344	EH344	100501	EMS140	076067																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
AMTYP4	=	000000	BIT15	=	100000	EH336	100305	EH337	100344	EH344	100501	EH353	071530	EH110	100126	EH111	100135	EMS141	076110																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
AOE	=	001000	BIT2	=	000004	EH110	100126	EH111	100135	EH114	100154	EH223	100203	EH256	100231	EH336	100305	EMS142	076136																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APASS	=	000000	BIT3	=	000010	EH114	100154	EH223	100203	EH256	100231	EH336	100305	EH337	100344	EH344	100501	EMS143	076153																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APE	=	100000	BIT4	=	000020	EH223	100203	EH256	100231	EH336	100305	EH337	100344	EH344	100501	EH353	071530	EMS144	076201																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APRIOR	=	000000	BIT5	=	000040	EH336	100305	EH337	100344	EH344	100501	EH353	071530	EH1	100107	EH110	100126	EMS145	076211																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APTCSU	=	000040	BIT6	=	000100	EH110	100126	EH111	100135	EH114	100154	EH223	100203	EH256	100231	EH336	100305	EMS146	076240																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APTENV	=	000001	BIT7	=	000200	EH111	100135	EH114	100154	EH223	100203	EH256	100231	EH336	100305	EH337	100344	EMS147	076257																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APTSIZ	=	000200	BIT8	=	000400	EH223	100203	EH256	100231	EH336	100305	EH337	100344	EH344	100501	EH353	071530	EMS15	072404																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
APTSPO	=	000100	BIT9	=	001000	EH336	100305	EH337	100344	EH344	100501	EH353	071530	EH1	100107	EH110	100126	EMS150	076274																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
ARGS	=	000004	BLNKS1	064563	EH110	100126	EH111	100135	EH114	100154	EH223	100203	EH256	100231	EH336	100305	EH337	100344	EMS151	076313																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ASNDA	001516	ASNDC	001514	ASWREG	=	000000	ATA	=	100000	ATESTN	=	000000	ATNMSK-	000377	ATNTBL	064666	AUNIT	000000	AUSWR	000000	AVECT1	=	120254	AVECT2	=	000000	A16	=	000400	A17	=	001000	BACK	=	000000	BADSCT	033152	BADTMO	005342	BAI	=	000010	BB00	=	000001	BB01	=	000002	BB02	=	000004	BB03	=	000010	BB04	=	000020	BB05	=	000040	BB06	=	000100	BB07	=	000200	BB08	=	000400	BB09	=	001000	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	064563	BLNKS2	064562	BLNKS3	064561	BLNKS4	064560	BOTADR	061024	BOTFLG	061026	BPTVEC	=	000014	BSE	=	100000	BUFFER	101360	BUFONE	101360	BUFTWO	102364	CC	=	004000	CH	=	002000	CHGADR	001330	CHRCNT	061027	CKSWR	=	104410	CLKADR	001520	CLKVCT	001522	CLR	=	000040	CLRSTS	045462	CMNSTA	007476	CMPBUF	035342	CMPERR	043460	CNSL01	064025	CNSL02	064035	CNSL03	064077	CNSL04	064106	CNSL07	064142	CNSL08	064304	CNSL09	064305	CNTCLR	045344	COMMA	063445	CONT	=	000100	CPSAVE	060150	CR	=	000015	CRLF	=	000200	CTLFG	001326	CYLSK	=	001777	DBCK	100000	DBEN	=	040000	DBL	=	002000	DCK	=	100000	DDISP	=	177570	DEBL	=	020000	DEVSEL	043672	DISPLA	001156	DISPRE	000174	DLT	=	100000	DMD	=	000001	DPE	=	000010	DPEHI	=	040000	DPELO	=	020000	DPR	=	000400	DRIVES	064520	DRQ	=	004000	DRVCLR	=	000010	DRVSTS	050700	DRY	=	000200	DSWR	=	177570	DTASTS	051502	DTE	=	010000	DTO	=	010000	DULPRT	=	024024	DVA	=	004000	DVC	=	000200	EARLY	065212	EBL	=	020000	ECH	=	000100	ECL	=	004000	ECRC	=	001000	EDT1	071534	EDT110	071552	EDT111	071554	EDT114	071556	EDT2	071544	EDT223	071560	EDT336	071570	EDT337	071600	EDT344	071610	EDT353	071622	ED1	101154	ED110	101162	ED111	101166	ED114	101174	ED223	101204	ED336	101214	ED337	101226	ED353	101240	ECC	=	000020	EFT1	071624	EFT110	071642	EFT111	071644	EFT114	071646	EFT2	071634	EFT223	071650	EFT336	071660	EFT337	071670	EFT344	071700	EFT353	071710	EF110	101332	EF111	101333	EF114	101335	EF336	101340	EF337	101344	EHT1	071410	EHT110	071432	EHT111	071436	EHT114	071442	EHT2	071422	EHT223	071446	EHT256	071460	EHT336	071472	EHT337	071504	EHT344	071516	EHT353	071530	EH1	100107	EH110	100126	EH111	100135	EH114	100154	EH223	100203	EH256	100231	EH336	100305	EH337	100344	EH344	100501	EH353	100637	EMS1	071712	EMS10	072246	EMS100	074464	EMS101	074512	EMS102	074540	EMS103	074567	EMS104	074576	EMS105	074625	EMS106	074654	EMS11	072310	EMS110	074673	EMS111	074722	EMS112	074733	EMS113	075017	EMS114	075047	EMS115	075073	EMS116	075076	EMS117	075126	EMS12	072321	EMS120	075150	EMS121	075171	EMS122	075234	EMS123	075271	EMS124	075334	EMS125	075366	EMS126	075431	EMS127	075474	EMS13	072362	EMS130	075533	EMS131	075571	EMS132	075631	EMS133	075655	EMS134	075700	EMS135	075742	EMS136	076005	EMS137	076032	EMS14	072373	EMS140	076067	EMS141	076110	EMS142	076136	EMS143	076153	EMS144	076201	EMS145	076211	EMS146	076240	EMS147	076257	EMS15	072404	EMS150	076274	EMS151	076313	EMS152	076337	EMS153	076363	EMS154	076405	EMS155	076421	EMS156	076441	EMS157	076457	EMS16	072426

EMS160	076474	EMS40	073215	EMT121	066636	EMT203	067522	EMT266	070274
EMS161	076511	EMS41	073244	EMT122	066646	EMT204	067534	EMT267	070306
EMS162	076541	EMS42	073272	EMT123	066656	EMT205	067552	EMT27	065550
EMS163	076541	EMS43	073321	EMT124	066666	EMT206	067562	EMT270	070324
EMS164	076565	EMS44	073350	EMT125	066676	EMT207	067572	EMT271	070340
EMS165	076646	EMS45	073423	EMT126	066706	EMT21	065470	EMT272	070356
EMS166	076662	EMS46	073466	EMT127	066720	EMT210	067604	EMT273	070374
EMS167	076707	EMS47	073515	EMT13	065412	EMT211	067612	EMT274	070412
EMS17	072456	EMS5	072104	EMT130	066732	EMT212	067626	EMT275	070430
EMS170	076734	EMS50	073544	EMT131	066744	EMT213	067642	EMT276	070442
EMS171	076763	EMS51	073573	EMT132	066756	EMT214	067652	EMT277	070456
EMS172	077016	EMS52	073621	EMT133	066770	EMT215	067672	EMT3	065336
EMS173	077036	EMS53	073631	EMT134	067002	EMT216	067704	EMT30	065560
EMS174	077053	EMS54	073643	EMT135	067014	EMT217	067714	EMT300	070474
EMS175	077061	EMS55	073665	EMT136	067026	EMT22	065500	EMT301	070514
EMS176	077107	EMS56	073714	EMT137	067040	EMT220	067724	EMT302	070536
EMS177	077136	EMS57	073771	EMT14	065424	EMT221	067734	EMT303	070550
EMS2	071761	EMS6	072134	EMT140	067050	EMT222	067744	EMT304	070562
EMS20	072516	EMS60	074017	EMT141	067060	EMT223	067754	EMT305	070574
EMS200	077160	EMS61	074032	EMT142	067070	EMT224	067764	EMT306	070606
EMS201	077175	EMS62	074061	EMT143	067100	EMT225	067766	EMT307	070616
EMS202	077243	EMS63	074077	EMT144	067110	EMT226	067770	EMT31	065570
EMS203	077273	EMS64	074125	EMT145	067120	EMT227	067772	EMT310	070636
EMS204	077306	EMS65	074143	EMT146	067130	EMT23	065510	EMT311	070650
EMS205	077334	EMS66	074211	EMT147	067140	EMT230	067774	EMT312	070662
EMS206	077346	EMS67	074261	EMT15	065432	EMT231	067776	EMT313	070674
EMS207	077373	EMS7	072201	EMT150	067150	EMT232	070000	EMT314	070714
EMS21	072541	EMS70	074306	EMT151	067160	EMT233	070002	EMT315	070726
EMS210	077420	EMS71	074320	EMT152	067172	EMT234	070004	EMT316	070740
EMS211	077431	EMS72	074333	EMT153	067204	EMT235	070006	EMT317	070752
EMS212	077447	EMS73	074343	EMT154	067222	EMT236	070010	EMT32	065600
EMS213	077472	EMS74	074360	EMT155	067240	EMT237	070012	EMT320	070762
EMS214	077515	EMS75	074377	EMT156	067252	EMT24	065520	EMT321	070772
EMS215	077533	EMS76	074405	EMT157	067264	EMT240	070014	EMT322	071002
EMS216	077554	EMS77	074436	EMT16	065440	EMT241	070016	EMT323	071010
EMS217	077604	EMTVEC=	000030	EMT160	067276	EMT242	070020	EMT324	071020
EMS22	072564	EMT1	065324	EMT161	067306	EMT243	070022	EMT325	071032
EMS220	077637	EMT10	065374	EMT162	067320	EMT244	070024	EMT326	071044
EMS221	077657	EMT100	066372	EMT163	067332	EMT245	070026	EMT327	071062
EMS222	077727	EMT101	066410	EMT164	067342	EMT246	070030	EMT33	065610
EMS223	077760	EMT102	066426	EMT165	067350	EMT247	070040	EMT330	071074
EMS224	100035	EMT103	066436	EMT166	067356	EMT25	065530	EMT331	071104
EMS23	072600	EMT104	066450	EMT167	067364	EMT250	070052	EMT332	071116
EMS24	072626	EMT105	066466	EMT17	065450	EMT251	070066	EMi 333	071130
EMS25	072655	EMT106	066476	EMT170	067372	EMT252	070074	EMT334	071146
EMS26	072672	EMT107	066506	EMT171	067404	EMT253	070102	EMT335	071164
EMS27	072713	EMT11	065400	EMT172	067414	EMT254	070120	EMT336	071204
EMS3	071776	EMT110	066526	EMT173	067424	EMT255	070130	EMT337	071214
EMS30	072724	EMT111	066540	EMT174	067434	EMT256	070140	EMT34	065620
EMS31	072734	EMT112	066546	EMT175	067446	EMT257	070150	EMT340	071224
EMS32	072763	EMT113	066554	EMT176	067454	EMT26	065540	EMT341	071236
EMS33	073012	EMT114	066570	EMT177	067462	EMT260	070162	EMT342	071244
EMS34	073040	EMT115	066576	EMT2	065330	EMT261	070174	EMT343	071256
EMS35	073111	EMT116	066606	EMT20	065460	EMT262	070214	EMT344	071270
EMS36	073140	EMT117	066616	EMT200	067464	EMT263	070224	EMT345	071302
EMS37	073167	EMT12	065404	EMT201	067476	EMT264	070234	EMT346	071312
EMS4	072041	EMT120	066626	EMT202	067510	EMT265	070254	EMT347	071326

EMT35	065630	ESRC	= 004000	ILRG76=	000076	PACACK=	000022	RMBAI	001342
EMT350	071340	FER	= 000020	IOTVEC-	000020	PAKACK=	000022	RMBAO	001416
EMT351	071356	FIND	= 000001	IPCK0	= 000001	PAR	= 000010	RMCS1	= 000000
EMT352	071366	FMT16	= 010000	IPCK1	= 000002	PAT	= 000020	RMCS1I	001336
EMT353	071372	FNCDTB	064566	IPCK2	= 000004	PDA	= 000400	RMCS1O	001412
EMT354	071402	FNCMSK=	000077	IPCK3	= 000010	PFECH	061072	RMCS2	= 000010
EMT36	065640	F0	= 000002	IR	= 000100	PFECH1	061102	RMCS2I	001346
EMT37	065650	F1	= 000004	IVC	= 010000	PFECH2	061170	RMCS2O	001422
EMT4	065344	F2	= 000010	LBC	= 002000	PFECH3	061206	RMCS3	= 000052
EMT40	065660	F3	= 000020	LBT	= 002000	PFECH4	061214	RMCS3I	001410
EMT41	065666	F4	= 000040	LF	= 000012	PGE	= 002000	RMCS3O	001464
EMT42	065676	GENBUF	035104	LODEV	064426	PGM	= 001000	RMDA	= 000006
EMT43	065710	GET	035770	LS	= 000004	PHA	= 000200	RMDAI	001344
EMT44	065720	GFTBUF	001336	LSC	= 004000	PIP	= 020000	RMDAO	001420
EMT45	065730	GETINX	001524	LST	= 000002	PIRQ	= 177772	RMDB	= 000022
EMT46	065740	GETSTS	035704	LSTRK	001334	PIRQVE=	000240	RMDBI	001360
EMT47	065750	GO	= 000001	MCLK	= 004000	PLFS	= 002000	RMDBO	001434
EMT5	065352	GTSWR	= 104407	MCPE	= 020000	PRIERR	036766	RMDC	= 000034
EMT50	065756	HCE	= 000200	MDF	= 000100	PR0	= 000000	RMDCI	001372
EMT51	065766	HCI	= 002000	MDPE	= 000400	PR1	= 000040	RMDCO	001446
EMT52	066000	HCRC	= 000400	MEDENB	001512	PR2	= 000100	RMDS	= 000012
EMT53	066016	HELP	101360	MFGFIL	103370	PR3	= 000140	RMDSI	001350
EMT54	066034	HT	= 000011	MI	= 000004	PR4	= 000200	RMDSO	001424
EMT55	066044	IAE	002000	MIXED	064676	PR5	= 000240	RMDT	= 000026
EMT56	066056	IBSAVE	060152	MOC	= 000400	PR6	= 000300	RMDTI	001364
EMT57	066074	IDXMSK-	000077	MOH	= 020000	PR7	= 000340	RMDTO	001440
EMT6	065360	IE	= 000100	MOL	= 010000	PS	= 177776	RMEC1	= 000044
EMT60	066104	ILF	= 000001	MRD	= 002000	PSEL	= 002000	RMEC1I	001402
EMT61	066122	ILF02	= 000002	MS	= 000040	PSW	= 177776	RMEC1O	001456
EMT62	066142	ILF24	= 000024	MSC	= 000002	PUT	036240	RMEC2	= 000046
EMT63	066156	ILF26	= 000026	MSDRVS	064326	PUTBUF	001412	RMEC2I	001404
EMT64	066160	ILF30	= 000030	MSE	= 100000	PUTINX	001553	RMEC2O	001460
EMT65	066200	ILF32	= 000032	MSER	= 000200	PWRVEC=	000024	RMER1	= 000014
EMT66	066220	ILF34	= 000034	MSGDRV	064342	QUES	063441	RMER1I	001352
EMT67	066232	ILF36	= 000036	MSHELP	063450	RCLSTS	047136	RMER1O	001426
EMT7	065366	ILF40	= 000040	MUR	= 001000	RD	= 000070	RMER2	= 000042
EMT70	066244	ILF42	= 000042	MWD	= 000010	RDCHR	= 104411	RMER2I	001400
EMT71	066254	ILF44	= 000044	MWP	= 000010	RDLIN	= 104412	RMER2O	001454
EMT72	066264	ILF46	= 000046	MXF	= 001000	RDOCT	= 104413	RMHR	= 000036
EMT73	066302	ILF54	= 000054	N	064554	RDY	= 000200	RMHRI	001374
EMT74	066320	ILF56	= 000056	NDTMSK=	115760	READY	007670	RMHRO	001450
EMT75	066330	ILF64	= 000064	NED	= 010000	RECAL	= 000006	RMLA	= 000020
EMT76	066350	ILF66	= 000066	NEM	= 004000	RESREG=	104415	RMLAI	001356
EMT77	066360	ILF74	= 000074	NONE	064547	RESVEC=	000010	RMLAO	001432
ENRGDT	065324	ILF76	= 000076	NOF	= 000000	REX	= 010000	RMMR1	000024
EQUALS	063432	ILR	= 000002	NOTAVL	064460	RG	= 040000	RMMR1I	001362
ERR	= 040000	ILRG50=	000050	NOTPRS	064443	RGDTPT	064676	RMMR1O	001436
ERRNMB	061022	ILRG52=	000052	NOTRM	064405	RH	= 000072	RMMR2	000040
ERROR	= 104000	ILRG54=	000054	NSA	= 100000	RIP	= 000020	RMMR2I	001376
ERRTYP	060154	ILRG56=	000056	OCC	= 100000	RELEASE=	000012	RMMR2O	001452
ERRVEC=	000004	ILRG60=	000060	OFD	= 000200	RMAS	= 000016	RMOF	= 000032
ERTY00	061030	ILRG62=	000062	OFFSET-	000014	RMASI	001354	RMOFI	001370
ERTY01	061035	ILRG64	000064	OM	= 000001	RMASO	001430	RMOFO	001444
ERTY02	061045	ILRG66=	000066	ONES	064736	RMBA	= 000004	RMR	= 000004
ERTY03	061054	ILRG70=	000070	OPE	= 020000	RMBAE	= 000050	RMSN	= 000030
ERTY04	061062	ILRG72	000072	GPI	= 020000	RMBAEI	001406	RMSNI	001366
ERTY05	061065	ILRG74	000074	OR	000200	RMBAEO	001462	RMSNO	001442

RMWJC = 000002	SW04 = 000020	TST25 025204	\$CMTAG 001114	\$LPERR 001124
RMWCI 001340	SW05 = 000040	TST26 026116	\$CM3 = 000000	\$MADR1 001254
RMWCO 001414	SW06 = 000100	TST27 026740	\$CM4 = 000005	\$MADR2 001260
RQA = 100000	SW07 = 000200	TST3 011130	\$CNTLC 062426	\$MADR3 001264
RQB = 040000	SW08 = 000400	TST30 027522	\$CNTLG 062440	\$MADR4 001270
RTC = 000016	SW09 = 001000	TST31 030672	\$CNTLU 062433	\$MAIL 00'222
R6 = 2000006	SW1 = 000002	TST4 011764	\$CPUOP 001250	\$MAMS1 001252
R7 = 2000007	SW10 = 002000	TST5 012554	\$CRLF 001217	\$MAMS2 001256
SADMSK= 000377	SW11 = 004000	TST6 013514	\$DBLK 056172	\$MAMS3 001262
SAVREG= 104414	SW12 = 010000	TST7 014334	\$DDW0 001306	\$MAMS4 001266
SA1 = 000001	SW13 = 020000	TYPBN = 104406	\$DDW1 001310	\$MBADR = 001102
SA16 000020	SW14 = 040000	TYPDS = 104405	\$DDW2 001312	\$MFLG 063342
SA2 00002	SW15 = 100000	TYPE = 104401	\$DDW3 001314	\$MNEW 062456
SA/ = 00'004	SW2 = 000004	TYPOC = 104402	\$DDW4 001316	\$MSGAD 001236
SAE = 00'010	SW3 = 000010	TYPON = 104404	\$DDW5 001320	\$MSGLG 001240
SC = 100000	SW4 = 000020	TYPOS = 104403	\$DDW6 001322	\$MSGTY 001222
SCOPE = 000004	SW5 = 000040	UNS = 040000	\$DDW7 001324	\$MSWR 062445
SCTMSG 063346	SW6 = 000100	UNTMSK= 000007	\$DEVCT 001232	\$MTYP1 001253
SCTMSK= 003700	SW7 = 000200	UNTOFF 064477	\$DEVM 001300	\$MTYP2 001257
SCO = C00100	SW8 = 000400	UNTON 064510	\$DOAGN 032132	\$MTYP3 001263
SC1 = 000200	SW9 = 001000	UPE = 020000	\$DTBL 056162	\$MTYP4 001267
SC2 = 000400	SYSTAT 064350	USE = 040000	\$ENDAD 032122	\$MXCNT 057446
SC3 = 001000	TADMSK= 177400	USRFIL 104376	\$ENDCT 031760	\$NULL 001170
SC4 = 002000	TAG = 020000	U0 = 000001	\$ENULL 032136	\$NWTST= 000001
SEARCH= 000030	TAGADR= 001114	U1 = 000002	\$ENV 001242	\$OCNT 056424
SECERR 037620	TAP = 040000	U2 = 000000	\$ENVM 001243	\$OMODE 056426
SEEK = 000004	TA1 = 000400	VV = 000100	\$EOP 031724	\$OVER 057432
SEKSTS 044104	TA16 = 010000	WC = 000040	\$EOPCT 031752	\$PASS 001230
SHUT 032142	TA2 = 001000	WCD = 000050	\$EOSP 031666	\$PASTM 001106
SHUT2 062470	TA4 = 002000	WCE = 040000	\$ERFLG 001117	\$POWER 063070
SIZCLK 036460	TA8 = 004000	WCEHI = 010000	\$ERMAX 001131	\$PURDN 062722
SKI = 040000	TBITVE= 000014	WCELO = 004000	\$ERROR 057560	\$PURMG 063056
SNGPRT= 020024	TIMOUT 036602	WCF = 000040	\$ERRPC 001132	\$PURUP 062774
STACK = 001100	TKVEC = 000060	WCH = 000052	\$ERRTB 001602	\$QUES 001216
STANDA 007020	TPVEC = 000064	WD = 000060	\$ERTIL 001126	\$RDCHR 062072
START 005432	TRAPVE= 000034	WH = 000062	\$ESCAP 001210	\$RDLIN 062162
START1 005422	TRE = 040000	WLE = 004000	\$ETABL 001242	\$RDOCT 062512
START2 005436	TRTVEC= 000014	WRL = 004000	\$ETEND 001326	\$RDSZ = 000010
STCDRV 055102	TST = 010000	XSIZ 006464	\$FATAL 001224	\$RESRE 055644
STKLMT= 177774	TSTNMB 061020	XXDP 001332	\$FFLG 063344	\$RM02 064366
STOP 057532	TSTPRP 032176	Y 064556	\$FILLC 001172	\$RM03 064373
STSD1 101250	TSTQUE 001466	ZEROS 065000	\$FILLS 001171	\$RM05 064400
STSD2 101266	TST1 010066	\$APTHD 001100	\$GDADR 001134	\$RTNAD 032134
STSD3 101306	TST10 015124	\$ATYC 063124	\$GDAT 001140	\$SAVRE 055606
STSD4 101320	TST11 016062	\$ATY1 063100	\$GET42 032112	\$SAVR6 063066
STSF 101351	TST12 016656	\$ATY3 063106	\$GT5WR 061620	\$SCOPE 056764
STSH1 100713	TST13 017470	\$ATY4 063116	\$GT42P 032106	\$SETUP= 000137
STSH2 100770	TST14 020302	\$AUTOB 001150	\$HD = 000000	\$STUP = 177777
STSH3 101057	TST15 021124	\$BASE 001276	\$HIBTS 001100	\$SVLAD 057376
STSH4 101115	TST16 021666	\$BDADR 001136	\$HIOCT 062612	\$SVPC = 000210
SWR 001154	TST17 022430	\$BDDAT 001142	\$ICNT 001120	\$SWR = 167400
SWREG 000176	TST2 010266	\$BELL 001212	\$ILLUP 063062	\$SWREG 001244
SWO 000001	TST20 023172	\$BIN 055754	\$INTAG 001151	\$SWRMK= 000000
SW00 = 000001	TST21 023476	\$CDW1 001302	\$ITEMB 001130	\$SW08T 057450
SW01 = 000002	TST22 024002	\$CDW2 001304	\$LF 001220	\$TESTN 001226
SW02 = 000004	TST23 024344	\$CHARC 056760	\$LFLG 063343	\$TIMES 001206
SW03 = 000010	TST24 024666	\$CKSWR 061530	\$LPADR 001122	\$TKB 001162

\$TKCNT	061220	\$TMP1	001176	\$TRAF-2	062654	\$TYPEC	056642	\$VECT2	001274
\$TKINT	061230	\$TMP2	001200	\$TRP =	000016	\$TYPEX	056762	\$XOFF =	000023
\$TKQEN-	061227	\$TMP3	001202	\$TRPAD	062666	\$TYPOC	056226	\$XON =	000021
\$TKOIN	061222	\$TMP4	001204	\$TSTM	001104	\$TYPON	056242	\$XTSTR	057006
\$TKOOL	061224	\$TN =	000032	\$TSTNM	001116	\$TYPOS	056202	\$\$GET4=	000000
\$TKQSR	061226	\$TPB	001166	\$TTYIN	U62416	\$UNIT	001234	\$\$SW08=	000032
\$TKS	001160	\$TPFLG	001173	\$TYPBN	055702	\$UNITM	001110	\$OFILL	056425
\$TKSRV	061300	\$TPS	001164	\$TYPDS	055756	\$USWR	001246	.\$X =	001100
\$TMPC	001174	\$TRAP	062614	\$TYPE	056430	\$VECT1	001272		

. ABS. 105404 000
000000 001
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 61696 WORDS (241 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 70 PAGES
CZRMNB.BIC,CZRMNB/C-CZRMNB.DOC,CZRMNB.SYSMAC/M

AENV	6-0	6-0												
AENVM	6-0	6-0												
AFATAL	6-0	6-0												
ALL	11-69	49-5#												
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	4-572#	4-583	25-311	25-326	25-341	25-356	25-360	26-117	26-121	50-77	50-78	50-81	50-82	50-85
	50-86													
APASS	6-0	6-0												
APE	4-750#													
APRIGR	6-0													
APTC SU	41-1	48-2#												
APTENV	41-1	43-1	48-2	48-2#										
APTSIZ	10-23	48-2#												
APTSPO	41-1	48-2	48-2#											
ARG5	13-43#	13-51	13-51#	13-63#	13-72#	13-76#	13-87#	13-93#	13-97#	13-99#	13-101#	13-108#	13-114#	13-118#
	13-120#	13-122#	13-126#	13-144#	13-152	13-152#	13-164#	13-173#	13-177#	13-189#	13-195#	13-199#	13-201#	13-203#
	13-211#	13-217#	13-221#	13-223#	13-225#	13-229#	13-242#	13-250	13-250#	13-262#	13-271#	13-275#	13-286#	13-292#
	13-296#	13-298#	13-300#	13-306#	13-312#	13-316#	13-318#	13-320#	13-335#	13-343	13-343#	13-355#	13-364#	13-368#
	13-379#	13-385#	13-389#	13-391#	13-393#	13-402#	13-408#	13-412#	13-419#	13-437#	13-460#	13-468	13-468#	13-480#
	13-489#	13-493#	13-504#	13-510#	13-514#	13-516#	13-518#	13-525#	13-531#	13-535#	13-537#	13-539#	13-543#	13-556#
	13-564	13-564#	13-576#	13-585#	13-589#	13-600#	13-606#	13-610#	13-612#	13-614#	13-620#	13-626#	13-630#	13-632#
	13-634#	13-649#	13-657	13-657#	13-669#	13-678#	13-682#	13-693#	13-699#	13-703#	13-705#	13-707#	13-716#	13-722#
	13-726#	13-733#	13-749#	13-773#	13-781	13-781#	13-793#	13-802#	13-806#	13-817#	13-823#	13-827#	13-829#	13-831#
	13-837#	13-843#	13-847#	13-849#	13-851#	13-867#	13-876	13-876#	13-888#	13-897#	13-901#	13-912#	13-918#	13-922#
	13-924#	13-926#	13-932#	13-938#	13-942#	13-944#	13-946#	13-962#	13-972	13-972#	13-984#	13-993#	13-997#	13-:08#
	13-:14#	13-:18#	13-:20#	13-:22#	13-:28#	13-:34#	13-:38#	13-:40#	13-:42#	13-:57#	13-:65	13-:65#	13-:79#	13-:88#
	13-:92#	13-:04#	13-:10#	13-:14#	13-:16#	13-:18#	13-:24#	13-:30#	13-:34#	13-:36#	13-:38#	13-:58#	13-:66	13-:66#
	13-:80#	13-:87#	13-:91#	13-:93#	13-:95#	13-<02#	13-<08#	13-<12#	13-<14#	13-<16#	13-<20#	13-<41#	13-<49	13-<49#
	13-<63#	13-<70#	13-<74#	13-<76#	13-<78#	13-<85#	13-<91#	13-<95#	13-<97#	13-<99#	13--03#	13--24#	13-=33	13-=33#
	13-=46#	13-=54#	13-=58#	13--60#	13-=62#	13-=68#	13-=73#	13-=77#	13-=79#	13--81#	13--85#	13->08	13->08#	13->21#
	13->28#	13->32#	13->34#	13->36#	13->54	13->54#	13->65#	13->72#	13->76#	13->78#	13->80#	13->96	13->96#	13-?09#
	13-?16#	13-?20#	13-?22#	13-?24#	13-?52	13-?52#	13-?65#	13-?72#	13-?76#	13-?78#	13-?80#	13-@01	13-@01#	13-@14#
	13-@21#	13-@25#	13-@27#	13-@29#	13-@49#	13-@57	13-@57#	13-@69#	13-@80#	13-@87#	13-@91#	13-@93#	13-@95#	13-A06#
	13-A19#	13-A25#	13-A29#	13-A31#	13-A33#	13-A37#	13-A56#	13-A64	13-A64#	13-A72#	13-A88#	13-A95#	13-A99#	13-B08#
	13-B12	13-B12#	13-B19#	13-B26#	13-B30#	13-B39#	13-B55#	13-B63	13-B63#	13-B77#	13-B84#	13-B88#	13-B90#	13-B92#
	13-C02#	13-C08#	13-C12#	13-C19#	13-C26#	13-C47#	13-C55	13-C55#	13-C77#	13-C84#	13-C88#	13-C90#	13-C92#	13-C99#
	13-D05#	13-D09#	13-D20#	13-D33#	13-D46#	13-D58#	13-D79	13-D79#	13-D90#	13-E10#	13-E19	13-E18#	13-E40#	13-E47#
	13-E51#	13-E53#	13-E55#	13-E62#	13-E68#	13-E72#	13-E79#	13-E91#	13-F12	13-F12#	13-F23#			
ASNDA	7-0#	16-299*	16-331	16-358*	16-359	16-361*	16-362*	16-363	16-365*	16-397				
ASNDC	7-0#	16-298*	16-330	16-366*	16-367	16-369*	16-396							
ASWREG	6-0	6-0												
A*A	4-552#	25-138	25-139	25-141	29-156	29-157	29-192	29-195	33-145	33-146	33-185	33-188	50-58	50-59
	50-60	50-63	50-64	50-67	50-68	50-69	50-70	50-71	50-72	50-73	50-74	50-75	50-76	50-79

	50-80	50-83	50-84	50-87	50-88										
ATESTN	6-0	6-0													
ATNMSK	4-589#	34-55													
ATNTBL	10-78	10-117	10-121	11-97	12-17	51-3#									
AUNIT	6-0	6-0													
AUSWR	6-0	6-0													
AVECT1	4-773#	6-0	6-0												
AVECT2	6-0	6-0													
BACK	13-43	13-43#	13-51	13-51#	13-63	13-63#	13-72	13-72#	13-76	13-76#	13-76#	13-87	13-87#	13-93	
	13-93#	13-97	13-97#	13-97#	13-99	13-99#	13-99#	13-101	13-101#	13-101#	13-101#	13-108	13-108#	13-114	
	13-118	13-118#	13-118#	13-120	13-120#	13-120#	13-122	13-122#	13-122#	13-122#	13-126	13-126#	13-144	13-152	
	13-152#	13-164	13-164#	13-173	13-173#	13-177	13-177#	13-177#	13-189	13-189#	13-189#	13-195	13-195#	13-199	
	13-199#	13-201	13-201#	13-201#	13-203	13-203#	13-203#	13-211	13-211#	13-217	13-217#	13-217#	13-221	13-221#	
	13-223	13-223#	13-223#	13-225	13-225#	13-225#	13-229	13-229#	13-242	13-242#	13-250	13-250#	13-262	13-262#	
	13-271	13-271#	13-275	13-275#	13-275#	13-286	13-286#	13-292	13-292#	13-296	13-296#	13-296#	13-298	13-298#	
	13-298#	13-300	13-300#	13-300#	13-306	13-306#	13-312	13-312#	13-316	13-316#	13-316#	13-318	13-318#	13-318#	
	13-320	13-320#	13-320#	13-335	13-335#	13-343	13-343#	13-355	13-355#	13-364	13-364#	13-368	13-368#	13-368#	
	13-379	13-379#	13-385	13-385#	13-389	13-389#	13-389#	13-391	13-391#	13-391#	13-393	13-393#	13-393#	13-402	
	13-402#	13-408	13-408#	13-412	13-412#	13-412#	13-419	13-419#	13-419#	13-437	13-437#	13-460	13-460#	13-468	
	13-468#	13-480	13-480#	13-489	13-489#	13-493	13-493#	13-493#	13-504	13-504#	13-510	13-510#	13-514	13-514#	
	13-514#	13-516	13-516#	13-516#	13-518	13-518#	13-518#	13-525	13-525#	13-531	13-531#	13-535	13-535#	13-535#	
	13-537	13-537#	13-537#	13-539	13-539#	13-539#	13-543	13-543#	13-556	13-556#	13-564	13-564#	13-576	13-576#	
	13-585	13-585#	13-589	13-589#	13-589#	13-600	13-600#	13-606	13-606#	13-610	13-610#	13-610#	13-612	13-612#	
	13-612#	13-614	13-614#	13-614#	13-620	13-620#	13-626	13-626#	13-630	13-630#	13-630#	13-632	13-632#	13-632#	
	13-634	13-634#	13-634#	13-649	13-649#	13-657	13-657#	13-669	13-669#	13-678	13-678#	13-682	13-682#	13-682#	
	13-693	13-693#	13-699	13-699#	13-703	13-703#	13-703#	13-705	13-705#	13-705#	13-707	13-707#	13-707#	13-716	
	13-716#	13-722	13-722#	13-726	13-726#	13-726#	13-733	13-733#	13-733#	13-749	13-749#	13-773	13-773#	13-781	
	13-781#	13-793	13-793#	13-802	13-802#	13-806	13-806#	13-806#	13-817	13-817#	13-823	13-823#	13-827	13-827#	
	13-827#	13-829	13-829#	13-829#	13-831	13-831#	13-831#	13-837	13-837#	13-843	13-843#	13-847	13-847#	13-847#	
	13-849	13-849#	13-849#	13-851	13-851#	13-851#	13-867	13-867#	13-876	13-876#	13-888	13-888#	13-897	13-897#	
	13-901	13-901#	13-901#	13-912	13-912#	13-918	13-918#	13-922	13-922#	13-922#	13-924	13-924#	13-924#	13-926	
	13-926#	13-926#	13-932	13-932#	13-938	13-938#	13-942	13-942#	13-942#	13-944	13-944#	13-944#	13-946	13-946#	
	13-946#	13-962	13-962#	13-972	13-972#	13-984	13-984#	13-993	13-993#	13-997	13-997#	13-997#	13-:08	13-:08#	
	13-:14	13-:14#	13-:18	13-:18#	13-:18#	13-:20	13-:20#	13-:20#	13-:22	13-:22#	13-:22#	13-:28	13-:28#	13-:34	
	13-:34#	13-:38	13-:38#	13-:38#	13-:40	13-:40#	13-:40#	13-:42	13-:42#	13-:42#	13-:57	13-:57#	13-:65	13-:65#	
	13-:79	13-:79#	13-:88	13-:88#	13-:92	13-:92#	13-:92#	13-:04	13-:04#	13-:10	13-:10#	13-:14	13-:14#	13-:14#	
	13-:16	13-:16#	13-:16#	13-:18	13-:18#	13-:18#	13-:24	13-:24#	13-:30	13-:30#	13-:34	13-:34#	13-:34#	13-:36	
	13-:36#	13-:36#	13-:38	13-:38#	13-:38#	13-:58	13-:58#	13-:66	13-:66#	13-:80	13-:80#	13-:87	13-:87#	13-:91	
	13-:91#	13-:91#	13-:93	13-:93#	13-:93#	13-:95	13-:95#	13-:95#	13-:02	13-:02#	13-:08	13-:08#	13-:12	13-:12#	
	13-<12#	13-<14	13-<14#	13-<14#	13-<16	13-<16#	13-<16#	13-<20	13-<20#	13-<41	13-<41#	13-<49	13-<49#	13-<63	
	13-<63#	13-<70	13-<70#	13-<74	13-<74#	13-<74#	13-<76	13-<76#	13-<76#	13-<78	13-<78#	13-<78#	13-<85	13-<85#	
	13-<91	13-<91#	13-<95	13-<95#	13-<95#	13-<97	13-<97#	13-<99	13-<99#	13-<99#	13-<99#	13-03	13-03#	13-24	
	13-24#	13-33	13-33#	13-46	13-46#	13-54	13-54#	13-58	13-58#	13-58#	13-60	13-60#	13-60#	13-62	
	13-62#	13-62#	13-68	13-68#	13-73	13-73#	13-77	13-77#	13-77#	13-79	13-79#	13-79#	13-81	13-81#	
	13-81#	13-85	13-85#	13->08	13->08#	13->21	13->21#	13->28	13->28#	13->32	13->32#	13->32#	13->34	13->34#	
	13->34#	13->36	13->36#	13->36#	13->54	13->54#	13->65	13->65#	13->72	13->72#	13->76	13->76#	13->76#	13->78	
	13->78#	13->78#	13->80	13->80#	13->80#	13->96	13->96#	13-?09	13-?09#	13-?16	13-?16#	13-?20	13-?20#	13-?20#	
	13-?22	13-?22#	13-?22#	13-?24	13-?24#	13-?24#	13-?52	13-?52#	13-?65	13-?65#	13-?72	13-?72#	13-?76	13-?76#	
	13-?76#	13-?78	13-?78#	13-?78#	13-?80	13-?80#	13-?80#	13-a01	13-a01#	13-a14	13-a14#	13-a21	13-a21#	13-a25	
	13-a25#	13-a25#	13-a27	13-a27#	13-a27#	13-a29	13-a29#	13-a29#	13-a49	13-a49#	13-a57	13-a57#	13-a69	13-a69#	
	13-a80	13-a80#	13-a87	13-a87#	13-a91	13-a91#	13-a91#	13-a93	13-a93#	13-a93#	13-a95	13-a95#	13-a95#	13-A06	
	13-A06#	13-A19	13-A19#	13-A25	13-A25#	13-A29	13-A29#	13-A29#	13-A31	13-A31#	13-A31#	13-A33	13-A33#	13-A33#	
	13-A37	13-A37#	13-A56	13-A56#	13-A64	13-A64#	13-A72	13-A72#	13-A88	13-A88#	13-A95	13-A95#	13-A99	13-A99#	
	13-A99#	13-B08	13-B08#	13-B08#	13-B12	13-B12#	13-B19	13-B19#	13-B26	13-B26#	13-B30	13-B30#	13-B30#	13-B39	
	13-B39#	13-B39#	13-B55	13-B55#	13-B63	13-B63#	13-B77	13-B77#	13-B84	13-B84#	13-B88	13-B88#	13-B88#	13-B90	
	13-B90#	13-B90#	13-B92	13-B92#	13-B92#	13-C02	13-C02#	13-C08	13-C08#	13-C12	13-C12#	13-C12#	13-C19	13-C19#	
	13-C19#	13-C26	13-C26#	13-C26#	13-C47	13-C47#	13-C55	13-C55#	13-C77	13-C77#	13-C84	13-C84#	13-C88	13-C88#	

DRVSTS	16-105	34-9#												
DRY	4-560#	24-176	25-38	25-41	25-42	29-195	31-127	31-128	34-30					
DSWR	4-484#	6-0	10-23											
DTASTS	13-99	13-120	13-201	13-223	13-298	13-318	13-391	13-419	13-516	13-537	13-612	13-632	13-705	13-733
	13-829	13-849	13-924	13-944	13-:20	13-:40	13-:16	13-:36	13-:93	13-<14	13-<76	13-<97	13-=60	13-=79
	13->34	13->78	13-?22	13-?78	13-a27	13-a93	13-A31	13-B90	13-C19	13-C90	13-D20	13-D33	13-D46	13-E53
	13-E79	16-200	35-17#											
DTE	4-569#	4-583	35-279	35-297	35-300									
DTO	4-607#													
DULPRT	4-646#													
DVA	4-490#	10-89	20-43	21-37	24-67	24-70	28-37	31-33	34-17	34-18				
DVC	4-680#	29-127	31-117	33-92	33-130	33-133	33-266	35-232	35-236	35-239	35-284	36-89	36-94	36-97
FARLY	52-107#													
EBL	4-623#													
ECH	4-575#	4-583	16-222	18-25	25-452	25-467	25-485	25-491	35-423	50-77	50-78	50-85	50-86	
ECI	4-651#	18-21	25-487	35-420										
ECRC	4-627#													
ED1	55-1	59-1#												
ED110	55-4	59-2#												
ED111	55-5	59-3#												
ED114	55-7	59-5#												
ED223	55-8	59-6#												
ED336	55-10	59-8#												
ED337	55-11	55-12	59-10#											
ED353	55-14	59-11#												
EDT1	8-3	8-6	8-9	8-12	8-15	8-18	8-24	8-27	8-30	8-33	8-36	8-39	8-42	8-45
	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81	8-84	8-87
	8-90	8-93	8-96	8-99	8-102	8-105	8-108	8-111	8-114	8-117	8-120	8-123	8-126	8-130
	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-160	8-163	8-166	8-169	8-172	8-175	8-178
	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212	8-215	8-218	8-236
	8-239	8-242	8-245	8-248	8-251	8-254	8-257	8-260	8-263	8-266	8-269	8-272	8-275	8-278
	8-281	8-284	8-287	8-290	8-293	8-296	8-299	8-302	8-305	8-308	8-311	8-314	8-317	8-320
	8-323	8-326	8-329	8-332	8-335	8-338	8-341	8-344	8-347	8-354	8-357	8-360	8-363	8-366
	8-369	8-372	8-375	8-378	8-381	8-384	8-390	8-393	8-396	8-399	8-402	8-405	8-408	8-411
	8-414	8-417	8-420	8-423	8-432	8-435	8-438	8-441	8-444	8-504	8-507	8-510	8-519	8-522
	8-525	8-531	8-534	8-537	8-540	8-543	8-546	8-549	8-552	8-556	8-559	8-563	8-566	8-569
	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600	8-603	8-606	8-609	8-612	8-615	8-618
	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642	8-645	8-648	8-651	8-654	8-657	8-660
	8-663	8-666	8-669	8-672	8-675	8-678	8-693	8-696	8-702	8-705	8-708	8-711	8-714	8-723
	55-1#													
EDT110	8-221	55-4#												
EDT111	8-224	8-227	55-5#											
EDT114	8-233	55-7#												
EDT2	8-426	8-429	8-513	8-516	55-2#									
EDT223	8-447	8-528	55-8#											
EDT336	8-351	8-681	8-687	8-690	55-10#									
EDT337	8-684	55-11#												
EDT344	8-699	55-12#												
EDT353	8-720	55-14#												
EECC	4-632#													
EF110	56-4	60-1#												
EF111	56-1	56-5	60-2#											
EF114	56-7	56-8	56-14	60-3#										
EF336	56-10	56-11	56-12	60-4#										
EF337	60-5#													
FFT1	8-3	8-6	8-9	8-12	8-15	8-18	8-24	8-27	8-30	8-33	8-36	8-39	8-42	8-45
	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81	8-84	8-87

EMS1	53-3	57-3#													
EMS10	53-7	53-11	57-10#												
EMS100	53-39	53-71	53-176	53-196	57-66#										
EMS101	53-40	53-59	53-174	53-197	57-67#										
EMS102	53-177	53-178	53-181	53-182	53-183	53-184	53-186	53-187	53-188	53-189	53-234	53-235	57-68#		
EMS103	53-56	53-57	53-135	53-141	53-196	53-197	53-198	53-203	53-204	53-205	53-206	53-207	53-208	53-209	
	53-210	53-211	53-213	53-214	53-215	53-219	53-220	57-69#							
EMS104	53-136	53-180	53-203	57-70#											
EMS105	53-204	57-71#													
EMS106	53-212	53-232	57-72#												
EMS11	53-12	53-15	53-16	53-17	53-18	53-19	53-20	53-21	53-22	53-23	53-24	53-25	53-26	53-27	
	53-28	53-29	53-30	53-31	53-32	53-33	53-43	53-177	57-11#						
EMS110	53-9	53-74	53-77	57-73#											
EMS111	53-74	53-226	57-74#												
EMS112	53-74	57-75#													
EMS113	53-75	53-76	53-78	53-127	57-77#										
EMS114	53-77	57-78#													
EMS115	53-44	53-45	53-48	53-50	53-51	53-52	53-54	53-55	53-60	53-61	53-63	53-66	53-67	53-70	
	53-73	53-77	53-107	53-108	53-109	53-110	53-134	53-137	53-139	53-140	53-142	53-179	53-182	53-183	
	53-185	53-186	53-187	53-188	53-189	53-190	53-192	53-194	53-195	53-201	53-205	53-221	53-222		
	53-223	53-232	53-234	57-79#											
EMS116	53-74	53-77	53-19	53-225	57-80#										
EMS117	53-77	53-143	53-148	53-149	53-178	53-185	53-201	53-225	53-228	53-229	53-230	53-231	57-81#		
EMS12	53-12	57-12#													
EMS120	53-78	57-82#													
EMS121	53-79	53-88	53-97	53-99	57-83#										
EMS122	53-80	53-89	57-84#												
EMS123	53-81	53-88	53-89	53-90	53-90	53-91	53-92	53-93	53-94	53-95	53-96	53-98	53-126	57-85#	
EMS124	53-56	53-82	53-91	53-101	57-86#										
EMS125	53-83	53-92	53-102	53-169	57-87#										
EMS126	53-84	53-93	53-103	57-88#											
EMS127	57-89#														
EMS13	53-13	53-42	53-216	57-13#											
EMS130	53-86	53-95	53-106	57-90#											
EMS131	53-87	53-96	53-104	57-91#											
EMS132	53-79	53-80	53-81	53-82	53-83	53-84	53-85	53-86	53-87	53-88	53-89	53-90	53-91	53-92	
	53-93	53-94	53-95	53-96	53-97	53-98	53-99	53-100	53-101	53-102	53-103	53-104	53-105	53-106	
	53-125	53-126	53-168	53-169	57-92#										
EMS133	53-79	53-80	53-81	53-82	53-83	53-84	53-85	53-86	53-87	53-125	57-93#				
EMS134	53-88	53-89	53-90	53-91	53-92	53-93	53-94	53-95	53-96	53-126	57-94#				
EMS135	53-97	53-98	57-95#												
EMS136	53-99	53-100	53-101	53-102	53-103	53-104	53-105	53-106	57-96#						
EMS137	53-85	53-94	53-100	57-97#											
EMS14	53-13	57-14#													
EMS140	53-44	53-52	53-60	53-107	53-117	53-137	53-186	53-192	57-98#						
EMS141	53-35	53-44	53-45	53-60	53-61	53-66	53-107	53-122	53-137	53-187	53-188	53-193	53-222	57-99#	
EMS142	53-47	53-65	53-108	53-140	53-184	53-202	57-100#								
EMS143	53-47	53-65	53-73	53-108	53-109	53-110	53-133	53-140	53-184	53-202	53-205	57-101#			
EMS144	53-36	53-73	53-109	53-110	53-205	57-102#									
EMS145	53-36	53-73	53-109	53-110	53-205	57-103#									
EMS146	53-109	53-185	53-190	53-201	57-104#										
EMS147	53-69	53-112	53-113	53-173	53-217	57-105#									
EMS15	53-13	53-42	53-216	57-15#											
EMS150	53-50	53-51	53-54	53-63	53-67	53-111	53-134	53-138	53-182	53-183	53-194	53-195	53-223	53-234	
	57-106#														
EMS151	53-69	53-112	53-113	53-173	53-217	57-107#									
EMS152	53-50	53-51	53-54	53-63	53-111	53-134	53-182	53-183	53-194	53-195	53-223	57-108#			

EMS153	53-111	57-109#												
EMS154	53-111	53-112	53-123	53-124	57-110#									
EMS155	53-112	53-124	57-111#											
EMS156	53-113	53-114	53-115	53-116	53-173	57-112#								
EMS157	53-113	53-123	57-113#											
EMS16	53-13	57-16#												
EMS160	53-114	53-115	53-116	57-114#										
EMS161	53-114	57-115#												
EMS162	57-116#													
EMS163	53-58	53-59	53-60	53-61	53-62	53-63	53-64	53-65	53-66	53-67	53-68	53-69	53-70	53-71
	53-72	53-73	53-117	53-143	57-117#									
EMS164	53-45	57-118#												
EMS165	53-56	53-57	53-71	53-72	53-172	53-196	53-197	53-198	57-120#					
FMS166	53-142	57-121#												
EMS167	53-8	53-49	53-68	53-123	53-124	53-168	53-169	53-170	53-175	53-235	57-122#			
EMS17	53-14	57-17#												
EMS170	53-9	57-123#												
EMS171	53-57	53-105	53-125	53-126	57-124#									
EMS172	53-215	57-125#												
EMS173	53-214	57-126#												
EMS174	53-216	57-127#												
EMS175	53-216	53-217	53-218	53-219	57-128#									
EMS176	57-129#													
EMS177	53-127	57-130#												
EMS2	53-4	53-5	57-4#											
EMS20	53-34	53-58	53-110	53-118	53-119	53-120	53-121	53-212	53-232	53-238	57-18#			
EMS200	53-128	57-131#												
EMS201	53-128	57-132#												
EMS202	53-122	53-130	53-131	53-132	53-133	53-134	53-136	53-180	57-133#					
EMS203	53-122	53-130	53-131	53-132	53-133	53-174	53-175	53-176	53-237	57-134#				
EMS204	53-131	57-135#												
EMS205	53-55	53-142	53-179	53-233	57-136#									
EMS206	53-55	53-142	53-170	53-179	53-233	57-137#								
EMS207	53-168	53-170	57-138#											
EMS21	53-15	57-19#												
EMS210	53-169	53-170	53-173	57-139#										
EMS211	53-170	57-140#												
EMS212	53-171	53-172	57-141#											
EMS213	53-171	57-142#												
EMS214	53-55	57-143#												
EMS215	53-236	57-144#												
EMS216	53-237	57-145#												
EMS217	53-237	57-146#												
EMS22	53-16	57-20#												
EMS220	53-179	57-147#												
EMS221	53-231	53-238	57-148#											
EMS222	53-233	57-149#												
EMS223	53-232	57-150#												
EMS224	53-121	57-151#												
EMS23	53-17	57-21#												
EMS24	53-18	57-22#												
EMS25	53-19	53-115	57-23#											
EMS26	53-20	53-116	53-173	57-24#										
EMS27	53-16	53-19	53-20	53-115	53-116	53-173	57-25#							
EMS3	53-4	53-76	57-5#											
EMS30	53-17	53-18	53-21	53-22	53-23	53-24	53-25	53-26	53-27	53-28	53-29	53-30	53-31	53-32
	53-33	53-43	53-122	53-130	53-131	53-132	53-133	53-144	53-145	53-146	53-147	53-174	53-175	53-176

EMT101	8-200	53-67#
EMT102	8-203	53-68#
EMT103	8-206	53-69#
EMT104	8-209	53-70#
EMT105	8-212	53-71#
EMT106	8-215	53-72#
EMT107	8-218	53-73#
EMT11	8-27	53-11#
EMT110	8-221	53-74#
EMT111	8-224	53-75#
EMT112	8-227	53-76#
EMT113	8-230	53-77#
EMT114	8-233	53-78#
EMT115	8-236	53-79#
EMT116	8-239	53-80#
EMT117	8-242	53-81#
EMT12	8-30	53-2#
EMT120	8-245	53-82#
EMT121	8-248	53-83#
EMT122	8-251	53-84#
EMT123	8-254	53-85#
EMT124	8-257	53-86#
EMT125	8-260	53-87#
EMT126	8-263	53-88#
EMT127	8-266	53-89#
EMT13	8-33	53-13#
EMT130	8-269	53-90#
EMT131	8-272	53-91#
EMT132	8-275	53-92#
EMT133	8-278	53-93#
EMT134	8-281	53-94#
EMT135	8-284	53-95#
EMT136	8-287	53-96#
EMT137	8-290	53-97#
EMT14	8-36	53-14#
EMT140	8-293	53-98#
EMT141	8-296	53-99#
EMT142	8-299	53-100#
EMT143	8-302	53-101#
EMT144	8-305	53-102#
EMT145	8-308	53-103#
EMT146	8-311	53-104#
EMT147	8-314	53-105#
EMT15	8-39	53-15#
EMT150	8-317	53-106#
EMT151	8-320	53-107#
EMT152	8-323	53-108#
EMT153	8-326	53-109#
EMT154	8-329	53-110#
EMT155	8-332	53-111#
EMT156	8-335	53-112#
EMT157	8-338	53-113#
EMT16	8-42	53-16#
EMT160	8-341	53-114#
EMT161	8-344	53-115#
EMT162	8-347	53-116#
EMT163	53-117#	

EMT164	8-354	53-118#
EMT165	8-357	53-119#
EMT166	8-360	53-120#
EMT167	8-363	53-121#
EMT17	8-45	53-17#
EMT170	8-366	53-122#
EMT171	8-369	53-123#
EMT172	8-372	53-124#
EMT173	8-375	53-125#
EMT174	8-378	53-126#
EMT175	8-381	53-127#
EMT176	8-384	53-128#
EMT177	53-129#	
EMT2	8-6	53-4#
EMT20	8-48	53-18#
EMT200	8-390	53-130#
EMT201	8-393	53-131#
EMT202	8-396	53-132#
EMT203	8-399	53-133#
EMT204	8-402	53-134#
EMT205	8-405	53-135#
EMT206	8-408	53-136#
EMT207	8-411	53-137#
EMT21	8-51	53-19#
EMT210	8-414	53-138#
EMT211	8-417	53-139#
EMT212	8-420	53-140#
EMT213	8-423	53-141#
EMT214	8-426	53-142#
EMT215	8-429	53-143#
EMT216	8-432	53-144#
EMT217	8-435	53-145#
EMT22	8-54	53-20#
EMT220	8-438	53-146#
EMT221	8-441	53-147#
EMT222	8-444	53-148#
EMT223	8-447	53-149#
EMT224	53-150#	
EMT225	53-151#	
EMT226	53-152#	
EMT227	53-153#	
FMT23	8-57	53-21#
EMT230	53-154#	
EMT231	53-155#	
EMT232	53-156#	
EMT233	53-157#	
EMT234	53-158#	
EMT235	53-159#	
EMT236	53-160#	
EMT237	53-161#	
EMT24	8-60	53-22#
EMT240	53-162#	
EMT241	53-163#	
EMT242	53-164#	
EMT243	53-165#	
EMT244	53-166#	
EMT245	53-167#	

EMT246	8-504	53-168#
EMT247	8-507	53-169#
EMT25	8-63	53-23#
EMT250	8-510	53-170#
EMT251	8-513	53-171#
EMT252	8-516	53-172#
EMT253	8-519	53-173#
EMT254	8-522	53-174#
EMT255	8-525	53-175#
EMT256	8-528	53-176#
EMT257	8-531	53-177#
EMT26	8-66	53-24#
EMT260	8-534	53-178#
EMT261	8-537	53-179#
EMT262	8-540	53-180#
EMT263	8-543	53-181#
EMT264	8-546	53-182#
EMT265	8-549	53-183#
EMT266	8-552	53-184#
EMT267	8-556	53-185#
EMT27	8-69	53-25#
EMT270	8-559	53-186#
EMT271	8-563	53-187#
EMT272	8-566	53-188#
EMT273	8-569	53-189#
EMT274	8-573	53-190#
EMT275	8-577	53-191#
EMT276	8-581	53-192#
EMT277	8-586	53-193#
EMT3	8-9	53-5#
EMT30	8-72	53-26#
EMT300	8-590	53-194#
EMT301	8-594	53-195#
EMT302	8-597	53-196#
EMT303	8-600	53-197#
EMT304	8-603	53-198#
EMT305	8-606	53-199#
EMT306	8-609	53-200#
EMT307	8-612	53-201#
EMT31	8-75	53-27#
EMT310	8-615	53-202#
EMT311	8-618	53-203#
EMT312	8-621	53-204#
EMT313	8-624	53-205#
EMT314	8-627	53-206#
EMT315	8-630	53-207#
EMT316	8-633	53-208#
EMT317	8-636	53-209#
EMT32	8-78	53-28#
EMT320	8-639	53-210#
EMT321	8-642	53-211#
EMT322	8-645	53-212#
EMT323	8-648	53-213#
EMT324	8-651	53-214#
EMT325	8-654	53-215#
EMT326	8-657	53-216#
EMT327	8-660	53-217#

EMT33	8-81	53-29#
EMT330	8-663	53-218#
EMT331	8-666	53-219#
EMT332	8-669	53-220#
EMT333	8-672	53-221#
EMT334	8-675	53-222#
EMT335	8-678	53-223#
EMT336	8-351	8-681 53-224#
EMT337	8-684	53-225#
EMT34	8-84	53-30#
EMT340	8-687	53-226#
EMT341	8-690	53-227#
EMT342	8-693	53-228#
EMT343	8-696	53-229#
EMT344	8-699	53-230#
EMT345	8-702	53-231#
EMT346	8-705	53-232#
EMT347	8-708	53-233#
EMT35	8-87	53-31#
EMT350	8-711	53-234#
EMT351	8-714	53-235#
EMT352	8-717	53-236#
EMT353	8-720	53-237#
EMT354	8-723	53-238#
EMT36	8-90	53-32#
EMT37	8-93	53-33#
EMT4	8-12	53-6#
EMT40	8-96	53-34#
EMT41	8-99	53-35#
EMT42	8-102	53-36#
EMT43	8-105	53-37#
EMT44	8-108	53-38#
EMT45	8-111	53-39#
EMT46	8-114	53-40#
EMT47	8-117	53-41#
EMT5	8-15	53-7#
EMT50	8-120	53-42#
EMT51	8-123	53-43#
EMT52	8-126	53-44#
EMT53	8-130	53-45#
EMT54	8-133	53-46#
EMT55	8-136	53-47#
EMT56	8-139	53-48#
EMT57	8-142	53-49#
EMT6	8-18	53-8#
EMT60	8-146	53-50#
EMT61	8-150	53-51#
EMT62	8-154	53-52#
EMT63	53-53#	
EMT64	8-160	53-54#
EMT65	8-163	53-55#
EMT66	8-166	53-56#
EMT67	8-169	53-57#
EMT7	8-21	53-9#
EMT70	8-172	53-58#
EMT71	8-175	53-59#
EMT72	8-178	53-60#

RESVEC	4-484#													
REX	4-624#													
RG	4-622#													
RGDTPT	52-3#													
RH	4-526#	13-105	13-208	13-522	13-;99	13-<82	13-=66	13->05	13->51	13->92	13-?48	13-?97	13-A11	13-B16
	13-C00	13-C96	13-E59	16-182										
RIP	4-508#													
RELEASE	4-505#													
RMA5	4-694#													
RMASI	7-0#	34-50	59-13											
RMASO	7-0#													
RMBA	4-766#	13-11*	13-12	13-83	13-185	13-282	13-375	13-500	13-596	13-689	13-813	13-908	13-:04	13-:00
	13-:76	13-<59	13-=43	13->14	13->61	13-?05	13-?61	13-?10	13-?76	13-A15	13-A84	13-B73	13-C73	13-D87
	13-E36	13-F19	16-68											
RMBAE	4-769#													
RMBAEI	7-0#													
RMBAEO	7-0#													
RMBAI	7-0#	13-430	13-742	26-32	26-34	31-44	59-14							
RMBAO	7-0#	13-40*	13-106*	13-141*	13-209*	13-239*	13-332*	13-457*	13-523*	13-553*	13-646*	13-770*	13-864*	13-959*
	13-:54*	13-:55*	13-<00*	13-<38*	13-<83*	13-=21*	13-=67*	13->04*	13->50*	13->90*	13-?47*	13-?96*	13-?46*	13-A12*
	13-A42*	13-A53*	13-B17*	13-B52*	13-B99*	13-C44*	13-C97*	13-D75*	13-E07*	13-E60*	13-F08*	16-61*	16-253	16-256*
	17-21	18-31	26-26	26-30										
RMCS1	4-690#	4-764#	10-89	13-8*	13-60	13-84	13-161	13-186	13-259	13-283	13-352	13-376	13-477	13-501
	13-573	13-597	13-666	13-690	13-790	13-814	13-885	13-909	13-981	13-:05	13-:76	13-:01	13-:77	13-<60
	13-=44	13->16	13->62	13-?06	13-?62	13-?11	13-?66	13-?77	13-A03	13-A16	13-A85	13-B74	13-C74	13-D88
	13-E37	13-F21	15-126	15-188	16-69	20-42	21-36	23-20	28-28					
RMCS1I	7-0#	16-209	24-67	24-69	24-71	24-88	24-90	24-93	24-104	24-108	24-110	24-122	24-124	24-126
	25-51	25-53	25-64	25-107	26-12	31-31	34-16	35-27	35-29	35-31	59-13			
RMCS10	7-0#	13-41*	13-55*	13-80*	13-105*	13-142*	13-156*	13-182*	13-208*	13-240*	13-254*	13-279*	13-304*	13-333*
	13-347*	13-372*	13-400*	13-458*	13-472*	13-497*	13-522*	13-554*	13-568*	13-593*	13-618*	13-647*	13-661*	13-686*
	13-714*	13-771*	13-785*	13-810*	13-835*	13-865*	13-880*	13-905*	13-930*	13-960*	13-976*	13-:01*	13-:26*	13-:55*
	13-:71*	13-:97*	13-:22*	13-:55*	13-:70*	13-:99*	13-<39*	13-<53*	13-<82*	13--22*	13-=37*	13--66*	13->05*	13->51*
	13->92*	13-?48*	13-?97*	13-?47*	13-?61*	13-?73*	13-?98*	13-A11*	13-A54*	13-A76*	13-B16*	13-B53*	13-B67*	13-C00*
	13-C45*	13-C67*	13-C96*	13-D76*	13-D81*	13-E08*	13-E30*	13-E59*	13-F09*	13-F14*	15-125*	15-185*	16-87*	16-119*
	16-153*	16-182*	16-304	17-26	25-28	26-48	35-320	35-340						
RMCS2	4-767#	10-84*	10-85*	10-87	12-58*	12-59*	13-13	13-14*	13-15	13-16*	20-41	21-35	28-27*	28-29
	30-16*	30-18*												
RMCS2I	7-0#	13-416	13-421	13-423	13-730	13-734	13-736	24-49	24-74	24-76	24-77	25-99	25-170	25-271
	25-314	25-329	25-344	25-455	26-28	31-53	35-50	35-63	35-65	35-67	35-438	35-440	35-442	35-492
	35-494	35-496	59-13											
RMCS20	7-0#													
RMCS3	4-770#													
RMCS3I	7-0#													
RMCS30	7-0#													
RMDA	4-691#	13-57	13-158	13-256	13-349	13-474	13-570	13-663	13-787	13-882	13-978	13-:73	13-:72	13-<55
	13- 39	13->11	13->57	13-?01	13-?57	13-?06	13-?63	13-A00	13-A80	13-B69	13-C69	13-D83	13-E32	13-F16
	15-187	16-64												
RMDAI	7-0#	26-137	59-14	59-16										
RMDAO	7-0#	13-38*	13-138*	13-236*	13-329*	13-454*	13-550*	13-643*	13-768*	13-860*	13-861*	13-869	13-955*	13-956*
	13-964	13-:51*	13-:51*	13-:60	13-<24*	13-<25	13-<34*	13-<43	13-=07*	13-=08	13-=17*	13-=99*	13->00*	13->44*
	13->45*	13->88*	13-?28*	13-?29	13-?34*	13-?43*	13-?44*	13-?84*	13-?85	13-?93*	13-?43*	13-?51	13-A50*	13-A58
	13-B48*	13-C42*	13-E04*	15-184*	16-58*	16-232*	16-233	16-235	16-255*	16-299	16-397*	17-24	26-60	26-61
	29-53	29-56	29-60	35-178	35-181	35-185								
RMDB	4-768#	13-17*	13-18	13-434	13-746	19-20								
RMDBI	7-0#	13-438	13-750											
RMDBO	7-0#													
RMDC	4-700#	13-58	13-159	13-257	13-350	13-475	13-571	13-664	13-788	13-883	13-979	13-:74	13-:73	13-<56

	13-40	13-12	13-58	13-202	13-258	13-207	13-264	13-A01	13-A81	13-B70	13-C70	13-D84	13-E33	13-F17
RMDCI	15-86	16-65												
RMDCO	7-0#	26-150	59-14	59-16										
	7-0#	13-37*	13-137*	13-235*	13-328*	13-453*	13-549*	13-642*	13-767*	13-859*	13-954*	13-:50*	13-:69	13-:70*
	13-:96*	13-:50*	13-:33*	13-:16*	13-:26	13-:89*	13-:90	13-:98*	13-:48*	13-:87*	13-:42*	13-:92*	13-:33*	13-:34
RMD5	13-242*	13-A49*	13-B47*	13-C41*	13-E03*	15-183*	16-57*	16-298	16-396*	17-23	26-59	29-49	35-174	
RMD5I	4-692#	10-86	10-109											
	7-0#	15-122	15-180	16-116	16-150	24-106	25-38	25-40	25-85	25-102	25-136	25-140	25-196	25-223
	25-358	26-92	29-91	29-146	29-155	29-162	29-166	29-168	29-177	29-181	29-183	29-192	29-194	29-196
	29-206	29-210	29-212	31-126	32-23	32-25	32-27	32-37	32-39	32-41	33-68	33-106	33-144	33-153
	33-157	33-159	33-169	33-173	33-174	33-185	33-187	33-189	33-201	33-203	33-205	33-216	33-220	33-222
	34-28	35-88	35-90	35-119	35-252	35-318	35-453	35-455	35-457	35-504	35-511	35-515	35-517	35-528
	35-532	35-534	35-544	35-548	35-550	36-34	36-40	36-44	36-46	36-56	36-60	36-61*	36-62	36-72
	36-76	36-78	59-13											
RMD5O	7-0#													
RMDT	4-697#	10-91	12-60	44-26										
RMDTI	7-0#	59-5	59-17											
RMDTO	7-0#													
RMEC1	4-704#													
RMEC1I	7-0#	18-32	59-14											
RMEC1O	7-0#													
RMEC2	4-705#	19-15												
RMEC2I	7-0#	18-53	19-14	31-94	34-94	59-15								
RMEC2O	7-0#													
RMER1	4-693#	24-145												
RMER1I	7-0#	13-C16	13-C21	13-C22	13-D17	13-D22	13-D23	13-D43	13-D48	13-D49	13-E76	13-E81	13-E82	16-222
	18-23	18-25	24-137	24-152	24-154	25-80	25-156	25-183	25-226	25-241	25-284	25-361	25-387	25-402
	25-417	25-470	25-489	25-492	26-89	26-105	26-120	26-132	27-24	29-30	29-37	29-66	29-140	29-143
	29-164	31-71	32-51	32-55	32-57	32-58	32-69	32-71	32-72	32-83	32-85	32-86	32-97	32-99
	32-100	32-111	32-113	32-114	33-24	33-29	33-33	33-35	33-45	33-47	33-49	33-61	33-63	33-65
	33-78	33-80	33-82	33-155	33-232	33-236	33-238	33-240	33-250	33-252	33-254	33-264	33-268	33-270
	34-41	35-41	35-45	35-47	35-83	35-85	35-87	35-128	35-131	35-133	35-135	35-145	35-147	35-149
	35-159	35-161	35-163	35-189	35-279	35-282	35-286	35-288	35-297	35-299	35-301	35-311	35-313	35-314
	35-348	35-352	35-354	35-356	35-365	35-367	35-369	35-378	35-380	35-382	35-391	35-393	35-395	35-413
	35-415	35-417	35-423	35-479	35-481	35-483	35-530	36-42	59-13					
RMER1O	7-0#	24-141												
RMER2	4-703#													
RMER2I	7-0#	13-B01	13-B03	13-B04	13-B32	13-B34	13-B35	13-D30	13-D35	13-D36	15-55	16-84	16-225	24-139
	25-82	25-200	25-256	25-432	27-27	29-32	29-85	29-88	29-99	29-127	29-130	29-179	29-208	31-116
	33-31	33-92	33-99	33-101	33-103	33-116	33-118	33-120	33-130	33-132	33-134	33-171	33-218	33-266
	34-103	35-43	35-111	35-114	35-116	35-203	35-207	35-232	35-236	35-238	35-240	35-250	35-254	35-256
	35-266	35-268	35-270	35-284	35-466	35-468	35-470	35-513	35-546	36-58	36-74	36-88	36-94	36-96
	36-98	36-108	36-110	36-112	59-13									
RMER2O	7-0#													
RMHR	4-701#													
RMHRI	7-0#													
RMHRO	7-0#													
RMLA	4-695#													
RMLAI	7-0#	59-16												
RMLAO	7-0#													
RMMR1	4-696#	13-A69	13-A79											
RMMR1I	7-0#	31-81	34-69	59-17										
RMMR1O	7-0#	13-A68*	13-A77*											
RMMR2	4-702#													
RMMR2I	7-0#	31-103	34-81	59-17										
RMMR2O	7-0#													
RMOF	4-699#	13-59	13-160	13-258	13-351	13-476	13-572	13-665	13-789	13-8E	13-980	13-:75	13-:74	13-:57

STCDRV	36-27#													
STKLMT	4-484#													
STUP	42-1	42-5#												
STSD1	55-1	55-2	55-8	55-10	55-11	55-12	59-13#							
STSD2	55-1	55-2	55-8	55-10	55-11	55-12	59-14#							
STSD3	59-16#													
STSD4	55-1	55-2	55-8	55-10	55-11	55-12	59-17#							
STSF	56-1	56-1	56-1	56-2	56-2	56-2	56-8	56-8	56-8	56-10	56-10	56-10	56-11	56-11
	56-11	56-12	56-12	56-12	60-7#									
STSH1	54-1	54-2	54-8	54-9	54-11	54-12	54-13	58-21#						
STSH2	54-1	54-2	54-8	54-9	54-11	54-12	54-13	58-23#						
STSH3	58-25#													
STSH4	54-1	54-2	54-8	54-9	54-11	54-12	54-13	58-26#						
SW0	4-484#													
SW00	4-484	4-484#												
SW01	4-484	4-484#												
SW02	4-484	4-484#												
SW03	4-484	4-484#												
SW04	4-484	4-484#												
SW05	4-484	4-484#												
SW06	4-484	4-484#												
SW07	4-484	4-484#												
SW08	4-484	4-484#												
SW09	4-484	4-484#												
SW1	4-484#													
SW10	4-484#													
SW11	4-484#													
SW12	4-484#													
SW13	4-484#	16-245	44-14											
SW14	4-484#													
SW15	4-484#													
SW2	4-484#													
SW3	4-484#													
SW4	4-484#													
SW5	4-484#													
SW6	4-484#													
SW7	4-484#													
SW8	4-484#													
SW9	4-484#													
SWR	6-0#	10-23	10-23	10-23*	10-23*	10-23*	10-28	16-245	42-1	42-1	42-1	42-1	42-1	42-1
	42-1*	42-1*	42-1*	42-1*	43-1	43-1	43-1	43-1	43-1	44-14	45-1	45-1	45-1*	48-1
	48-1*													
SWREG	5-1#	10-23	10-28	45-1	45-1	45-1								
SYSTAT	10-76	49-23#												
TA1	4-537#													
TA16	4-533#	12-64												
TA2	4-536#	12-64												
TA4	4-535#	12-56												
TA8	4-534#													
TADMSK	4-547#	26-66												
TAG	4-664#													
TAGADR	5-9#	6-0												
TAP	4-641#													
TBITVE	4-484#													
TIMOUT	13-68	13-91	13-112	13-169	13-193	13-215	13-267	13-290	13-310	13-360	13-383	13-406	13-485	13-508
	13-529	13-581	13-604	13-624	13-674	13-697	13-720	13-798	13-821	13-841	13-893	13-916	13-936	13-989
	13-:12	13-:32	13-:84	13-:08	13-:28	13-:85	13-<06	13-<68	13-<89	13-=53	13--72	13->26	13->70	13-?14

UNTOFF	10-120	49-31#												
UNTON	10-131	49-32#												
UPE	4-729#	25-270												
USE	4-686#	13-D11	17-29											
USRFIL	16-253	16-256	16-266*	16-274*	16-325	61-12#								
VV	4-561#	15-122	16-116	25-196	29-91	29-156	29-157	29-206	29-211	32-23	32-26	33-106	33-145	33-14#
	33-169	33-175	34-29	35-70	35-119	35-505	35-506	35-544	35-549	36-35	36-36	36-56	36-61	
WC	4-631#	31-82	34-70											
WCD	4-517#													
WCE	4-728#	13-416	13-422	13-730	13-735	25-169	50-77	50-78						
WCEHI	4-753#													
WCELO	4-754#													
WCF	4-576#	4-583	25-240	35-479	35-482									
WCH	4-518#	13-304	13-400	13-618	13-714	13-835	13-930	13-:26	13-:22					
WD	4-521#													
WH	4-522#	13-41	13-80	13-142	13-182	13-240	13-279	13-333	13-372	13-458	13-497	13-554	13-593	13-647
	13-686	13-771	13-810	13-865	13-905	13-960	13-:01	13-:55	13-:97	13-:56	13-:70	13-<39	13-<53	13-:22
	13-:37	13-@47	13-@73	13-A54	13-A76	13-B53	13-B67	13-C45	13-C67	13-D76	13-D81	13-E08	13-E30	13-F09
	13-F14	35-342												
WLE	4-570#	4-583	25-221	25-225	25-238	25-253	25-268	35-279	35-311	35-315	35-323	50-81	50-82	
WRL	4-556#	25-223	35-318											
XSIZ	10-70#	11-71	11-101											
XXDP	7-0#	10-33*	10-36*	10-37	10-39*	10-44	10-55	10-124	10-126					
Y	49-36#													
ZEROS	13-45	13-146	13-244	13-337	13-775	13-871	13-966	13-:59	13-B57	13-C49	13-E12	52-38#		

SSCMRE	5-10#													
SSCMTM	5-10#	6-0	6-0	6-0	6-0	6-0								
SSESCA	4-484#													
SSNEWT	4-484#	13-1	13-32	13-134	13-232	13-325	13-450	13-546	13-639	13-763	13-856	13-951	13-:47	13-:47
	13-<30	13-=13	13-=95	13->41	13->85	13-?40	13-?90	13-@39	13-A46	13-B44	13-C34	13-D97		
SSSET	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1	47-1#
SSSETM	10-23	10-23#												
SSSKIP	4-484#													
.SACT1	4-476#	5-5												
.SAPTB	4-476#	6-0	6-0#											
.SAPTH	4-476#	5-8												
.SAPTY	4-476#	48-2												
.SCATC	4-472#	5-1												
.SCMTA	4-473#	5-10												
.SEOP	4-473#	14-20												
.SERRO	4-473#	43-1												
.SPOWE	4-475#	48-1												
.SRDDE	4-474#													
.SRDOC	4-474#	46-1												
.SREAD	4-474#	45-1												
.SSAVE	4-475#	37-1												
.SSCOP	4-473#	42-1												
.SSIZE	4-475#													
.STRAP	4-475#	47-1												
.STYPB	4-474#	38-1												
.STYPD	4-474#	39-1												
.STYPE	4-473#	41-1												
.STYPO	4-474#	40-1												
.EQUAT	4-472#	4-484												
.HEADE	4-472#	4-480												
.SETUP	4-472#	4-775												
.SWRHI	4-472#	4-481												
.SWRLO	4-472#	4-481#	4-482											
CALCLR	4-187#													
CALSUB	4-201#	13-43	13-51	13-63	13-72	13-76	13-87	13-93	13-97	13-99	13-101	13-108	13-114	13-118
	13-120	13-122	13-126	13-144	13-152	13-164	13-173	13-177	13-189	13-195	13-199	13-201	13-203	13-211
	13-217	13-221	13-223	13-225	13-229	13-242	13-250	13-262	13-271	13-275	13-286	13-292	13-296	13-298
	13-300	13-306	13-312	13-316	13-318	13-320	13-335	13-343	13-355	13-364	13-368	13-379	13-385	13-389
	13-391	13-393	13-402	13-408	13-412	13-419	13-437	13-460	13-468	13-480	13-489	13-493	13-504	13-510
	13-514	13-516	13-518	13-525	13-531	13-535	13-537	13-539	13-543	13-556	13-564	13-576	13-585	13-589
	13-600	13-606	13-610	13-612	13-614	13-620	13-626	13-630	13-632	13-634	13-649	13-657	13-669	13-678
	13-682	13-693	13-699	13-703	13-705	13-707	13-716	13-722	13-726	13-733	13-749	13-773	13-781	13-793
	13-802	13-806	13-817	13-823	13-827	13-829	13-831	13-837	13-843	13-847	13-849	13-851	13-867	13-876
	13-888	13-897	13-901	13-912	13-918	13-922	13-924	13-926	13-932	13-938	13-942	13-944	13-946	13-962
	13-972	13-984	13-993	13-997	13-:08	13-:14	13-:18	13-:20	13-:22	13-:28	13-:34	13-:38	13-:40	13-:42
	13-:57	13-:65	13-:79	13-:88	13-:92	13-:04	13-:10	13-:14	13-:16	13-:18	13-:24	13-:30	13-:34	13-:36
	13-:38	13-:58	13-:66	13-:80	13-:87	13-:91	13-:93	13-:95	13-<02	13-<08	13-<12	13-<14	13-<16	13-<20
	13-<41	13-<49	13-<63	13-<70	13-<74	13-<76	13-<78	13-<85	13-<91	13-<95	13-<97	13-<99	13-=03	13-=24
	13-=33	13=-46	13-=54	13=-58	13-=60	13-=62	13-=68	13-=73	13=-77	13-=79	13=-81	13=-85	13->08	13->21
	13->28	13->32	13->34	13->36	13->54	13->65	13->72	13->76	13->78	13->80	13->96	13-?09	13-?16	13-?20
	13-?22	13-?24	13-?52	13-?65	13-?72	13-?76	13-?78	13-?80	13-@01	13-@14	13-@21	13-@25	13-@27	13-@29
	13-@49	13-@57	13-@69	13-@80	13-@87	13-@91	13-@93	13-@95	13-A06	13-A19	13-A25	13-A29	13-A31	13-A33
	13-A37	13-A56	13-A64	13-A72	13-A88	13-A95	13-A99	13-B08	13-B12	13-B19	13-B26	13-B30	13-B39	13-B55
	13-B63	13-B77	13-B84	13-B88	13-B90	13-B92	13-C02	13-C08	13-C12	13-C19	13-C26	13-C47	13-C55	13-C77
	13-C84	13-C88	13-C90	13-C92	13-C99	13-D05	13-D09	13-D20	13-D33	13-D46	13-D58	13-D79	13-D90	13-E10
	13-E18	13-E40	13-E47	13-E>1	13-E53	13-E55	13-E62	13-E68	13-E72	13-E79	13-E91	13-F12	13-F23	

