

RM05/3/2

RM05/3/2 DSKLS TST2
CZRMQAO

AH-F934A-MC
FICHE 1 OF 1

JUN 1980
COPYRIGHT © 1980
MADE IN USA



The main body of the document is a microfiche card containing a grid of approximately 20 columns and 20 rows of data. Each cell in the grid contains a small, dense block of text, likely representing a single page of a document. The text is too small to be legible in this image, but it appears to be organized in a structured, tabular format. The data is arranged in a regular grid pattern across the entire page.

.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
45
46
47
48
49
50
51

IDENTIFICATION

PRODUCT CODE: AC-F933A-MC
PRODUCT NAME: CZRMQA0 RM05/3/2 DSKLS TST 2
DATE CREATED: APRIL 1980
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 TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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 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

- 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. OPERATOR INTERFACE
 - 1. PROGRAM ID
 - 2. PROGRESS REPORTS
 - 3. PERFORMANCE REPORTS
 - 4. PROGRAM HALTS
 - 5. ERROR REPORTS
- 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 DISKLESS DIAGNOSTIC IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL AND DIAGNOSTIC MEANS TO VERIFY THE OPERABILITY OF THE DISK SUBSYSTEM EXCLUDING AND INDEPENDENTLY OF THE STORAGE MODULE DRIVE. IN PARTICULAR, THE PROGRAM SERVES THE FOLLOWING PURPOSES:

TO DETECT ERRORS AND FAULTS IN THE RM MASSBUS ADAPTER;

TO RESOLVE HARDWARE FAILURES IN RM TO A FIELD REPLACEABLE MODULE OR MODULES.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM05/3/2 DISK SUBSYSTEM, EXCLUDING THE STORAGE MODULE DISK DRIVE AND THE MASSBUS CONTROLLER.

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 DISKLESS DIAGNOSTIC:

- PDP-11 PROCESSOR
- 16K MEMORY
- KW11-L OR LW11-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)

2.2 MEDIA REQUIREMENTS

NONE

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM05/3/2 DISKLESS DIAGNOSTIC, PART 1

3.0 OPERATING PROCEDURE

3.1 LOADING

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

THE PROGRAM MAY BE LOADED BY EITHER PAPER TAPE, USING THE STANDARD PAPER TAPE LOADING PROCEDURE, OR XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

3.2 SWITCH OPTIONS

THE FOLLOWING SWITCH OPTIONS ARE INVOKED WHEN THE APPROPRIATE SWITCH IS ON.

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 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 STARTS AT LOCATION 200 WHICH USES DEFAULT VALUES OF PARAMETERS AND PROVIDES MAXIMUM TESTING WITH THE SWITCH REGISTER EQUAL TO ZERO.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

THE PROGRAM TYPES ITS NAME AND MAINDEC NUMBER THE FIRST TIME IT IS STARTED AFTER BEING LOADED.

4.2 PROGRESS REPORTS

AN END OF PASS REPORT OCCURRS EACH TIME THE PROGRAM IS EXECUTED FOR ALL ADAPTERS IN THE TEST QUE. THE END OF PASS REPORT INCLUDES A MESSAGE AND AN ERROR SUMMARY.

4.3 PERFORMANCE REPORT

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

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

4.4 PROGRAM HALTS

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

4.5 ERROR REPORTS

THE RM05/3/2 DISKLESS DIAGNOSTIC PROVIDES COMPREHENSIVE ERROR REPORTS INTENDED TO (1) AID IN FAULT RESOLUTION AND (2) MINIMIZE REFERENCES TO PROGRAM LISTINGS.

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE UNIT NUMBER BEING TESTED, 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: SEVERAL LINES OF TEXT WHICH GIVE A COMPREHENSIVE DESCRIPTION OF THE ERROR, AND A LIST OF FAILING MODULES IN ORDER OF DECREASING PROBABILITY. 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.

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RM05/3/2 DISKLESS DIAGNOSTIC IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET.

5.2 DUAL PORT CONFIGURATIONS

THE RM05/3/2 DISKLESS DIAGNOSTIC IS NOT EXECUTABLE ON RM05/3/2 SUBSYSTEMS HAVING THE DUAL PORT OPTION UNLESS THE DUAL PORT SWITCH IS SET TO THE APPROPRIATE PORT (A OR B) AND NOT TO THE PROGRAMMABLE POSITION (A/B).

5.3 MEMORY PARITY HARDWARE

MEMORY PARITY HARDWARE WILL NOT BE USED DURING THE EXECUTION OF THE RM05/3/2 DISKLESS DIAGNOSTIC.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE WILL NOT BE USED DURING THE RM05/3/2 DISKLESS DIAGNOSTIC.

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

5.5 ACT11, APT11 COMPATIBILITY

THE RM05/3/2 DISKLESS DIAGNOSTIC PROGRAM 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 DISKLESS DIAGNOSTIC PROGRAM IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES.

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT REQUIRED TO BE COMPATIBLE WITH ANY OPERATING SYSTEM.

6.0 TEST DESCRIPTION

THE PROGRAM IS DESIGNED IN A BOTTOM UP MANNER SUCH THAT EACH TEST GENERALLY USES A MORE COMPLEX SUBSET OF HARDWARE THAN THE PREVIOUS TEST.

MODULE CALLOUT IS PREDICATED ON THE ASSUMPTION THAT EARLIER TESTS HAVE BEEN COMPLETED WITHOUT ERROR AND THAT ERRORS ARE DUE TO SINGLE, NONTRANSIENT HARDWARE FAILURES.

THE 'RM05/3/2 DISKLES DIAGNOSTIC' CAN BE EXECUTED USING AN RH70 OR AN RH11 MASSBUS CONTROLLER.

UNLESS SPECIFIED BY THE OPERATOR OR BY THE ENVIRONMENT TABLE THE TEST IS REPEATED FOR EACH POSSIBLE DEVICE STARTING WITH DEVICE 0.

THE MODULES WHICH MAY BE CALLED OUT DURING THE EXECUTION OF THE TEST ARE AS FOLLOWS:

- IF
- CS
- DS
- MASSBUS MODULE

THE RADIAL MODULE (RD) IS NOT TESTED BY THIS PROGRAM.

TRANSFER TEST

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

PURPOSE:

TO VERIFY THAT THE RM05/3/2 CAN COMPLETE A REGISTER TRANSFER ON THE MASSBUS, AND, IN PARTICULAR, TO VERIFY THAT 'TRANSFER' IS NOT STUCK IN AN INACTIVE STATE.

PROCEDURE:

THE PROGRAM WRITES AND READS REMOTE REGISTERS FOR THE SELECTED DEVICE. REGISTER CONTENTS AND PARITY ERRORS ARE IGNORED, AND THE TEST FAILS IF A 'NONEXISTENT DEVICE ERROR' OR BUS TIMEOUT OCCURS FOR EVERY REGISTER ACCESS. IF THE TEST FAILS THE PROGRAM JUMPS TO THE END OF PASS HANDLER WHICH SELECTS THE NEXT DEVICE TO BE TESTED.

PROBABLE FAULT:

THE TEST FAILS IF THE SELECTED DEVICE IS NONEXISTENT OR IS SWITCHED TO THE PROGRAMMABLE POSITION OR TO THE ALTERNATE PORT. THE FOLLOWING FAULTS ARE APPLICABLE ONLY WHEN THE DEVICE IS PRESENT AND IS SWITCHED TO THE APPROPRIATE PORT.

1. IF MODULE
2. ASYNCHRONOUS MASSBUS MODULE
3. CS MODULE

CTOD TEST

PURPOSE:

TO VERIFY THAT DATA CAN BE TRANSFERRED TO AND FROM THE RM05/3/2 USING THE CONTROL BUS AND, IN PARTICULAR, TO VERIFY THAT 'CONTROLLER TO DEVICE' HAS NOT FAILED.

PROCEDURE:

THE TEST WRITES ONES IN REMOTE REGISTERS THEN READS EACH REGISTER WHICH WILL WRITE ZEROS IN THE REGISTER IF 'IF3 CTOD HOLD H' IS STUCK AT ONE. THE TEST THEN READS AS MANY REMOTE REGISTERS AS ARE NECESSARY TO OBTAIN ONE OR MORE ONE BITS.

PROBABLE FAULT:

1. IF MODULE
2. ASYNCHRONOUS MASSBUS MODULE

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

MASSBUS INITIALIZE TEST

PURPOSE:

TO VERIFY THAT THE MASSBUS ADAPTER IS BEING INITIALIZED BY THE MASS BUS.

PROCEDURE:

USING CONTROLLER CLEAR TO INITIALIZE THE SELECTED UNIT, THIS TEST THEN READS MASSBUS ADAPTER REGISTERS TO VERIFY THAT AT LEAST ONE BIT IS CLEARED. MASSBUS ADAPTER REGISTERS ARE PRESET TO A NON ZERO VALUE PRIOR TO CONTROLLER CLEAR.

PROBABLE FAULT:

- 1. ASYNCHRONOUS MASSBUS MODULE
- 2. IF MODULE
- 3. CS MODULE

CLEAR STUCK ACTIVE TEST

PURPOSE:

TO VERIFY THAT 'MBA CLR L' ON THE CS MODULE IS NOT STUCK IN AN ACTIVE STATE.

PROCEDURE:

CONTROLLER CLEAR IS USED TO INITIALIZE THE SELECTED UNIT, AFTER WHICH 1'S ARE WRITTEN IN ERROR REGISTERS 1 AND 2 AND MAINTENANCE REGISTER 1. IF ANY 1 BITS CAN BE READ BACK THE TEST IS OK, ELSE, 'MBA CLR L' IS PROBABLY STUCK ACTIVE.

PROBABLE FAULT:

- 1. CS MODULE
- 2. IF MODULE
- 3. ASYNCHRONOUS MASSBUS MODULE

TRISTATE TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE PATH TO AND FROM THE MASSBUS ADAPTER

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

TRI-STATE REGISTER BUS IS NOT STUCK AT ONE OR ZERO AND THAT EACH BIT POSITION IS INDEPENDENT.

PROCEDURE:

THIS TEST PRESETS MASSBUS ADAPTER REGISTERS TO A NONZERO VALUE, THEN, ASSUMING THE REGISTERS ARE PRESET, IT CLEARS THEM USING A MOVE INSTRUCTION. THE TEST THEN READS AS MANY REGISTERS AS IS NECESSARY TO OBTAIN ONE OR MORE ZEROS FROM EACH BIT POSITION.

THE TEST CLEARS MASSBUS ADAPTER REGISTERS, THEN, ASSUMING THE REGISTERS ARE CLEARED, IT LOADS THEM WITH ONES AND READS AS MANY REGISTERS AS IS NECESSARY TO OBTAIN ONE OR MORE ONE BITS IN EACH BIT POSITION.

FINALLY, THE TEST WRITES A SINGLE ONE BIT PATTERN IN BIT 0 OF SELECTED REMOTE REGISTERS AND VERIFIES THAT THE PATTERN CAN BE READ BACK. THE ONE BIT IS SHIFTED AND THE TEST REPEATED FOR ALL BIT POSITIONS.

PROBABLE FAULT:

1. ASYNCHRONOUS MASSBUS MODULE
2. IF MODULE
3. CS MODULE
4. DS MODULE

REGISTER SELECT TEST

PURPOSE:

TO VERIFY THAT THE REGISTER SELECT LINES ARE NOT IN A STUCK POSITION.

PROCEDURE:

EACH REGISTER SELECT LINE IS TESTED BY WRITING ZEROS IN THOSE DEVICE REGISTERS FOR WHICH THE LINE MUST BE ZERO, THEN WRITING ONES IN THOSE DEVICE REGISTERS FOR WHICH THE LINE MUST BE ONE. THE ZERO REGISTER IS READ BACK AND IF THE SELECT LINE IS STUCK AT ZERO, THE ZERO REGISTER WILL CONTAIN ONES. THE PROCESS IS REPEATED TO DETECT A STUCK AT ONE FAULT, EXCEPT IN THIS CASE, THE ONES REGISTER IS WRITTEN FIRST.

REGISTER SELECT LINES 1, 2, 4 AND 8 ARE TESTED IN THIS MANNER; SELECT LINE 16 IS EXPLICITLY TESTED IN THE "ILR TEST".

PROBABLE FAULT:

40C
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

- 1. IF MODULE
- 2. ASYNCHRONOUS MASSBUS MODULE

DRIVE TYPE TEST

PURPOSE:
TO TEST THE 'DRIVE TYPE' REGISTER, RMDT.

PROCEDURE:
THE PROGRAM READS RMDT AND VERIFIES THAT THE RESULT
CORRESPONDS TO A SINGLE OR DUAL PORT RM05, RM03 OR RM02 DRIVE

PROBABLE FAULT:
1. IF MOULE

DEVICE AVAILABLE TEST

PURPOSE:
TO VERIFY THAT DEVICE AVAILABLE STATUS IS SET.

PROCEDURE:
THE PROGRAM TESTS 'DVA',BIT 11 OF RMCS1

PROBABLE FAULT:
1. IF MODULE

SEARCH TIMEOUT TEST

PURPOSE:
TO VERIFY THAT THE SEARCH TIMEOUT ONE SHOT SETS 'OPI',
EXCEPT WHEN 'SEARCH TO DISABLE' IS ACTIVE.

PROCEDURE:
WITH SEARCH TIMEOUT DISABLED, THE TEST EXECUTES A DATA
COMMAND TO THE POINT WHERE 'P ENABLE SEARCH' IS ASSERTED.
AFTER WAITING A SUFFICIENT PERIOD AND VERIFYING THAT OPI

457 IS NOT SET, THE TEST ENABLES SEARCH TIMEOUT AND VERIFIES
458 THAT OPI SETS.
459

460 PROBABLE FAULT:

461 1. CS MODULE
462

463 NOTE: IT IS ASSUMED THAT THE 'SET OPI TEST' HAS
464 ALREADY PASSED, THUS MAKING THE IF MODULE
465 AN IMPROBABLE FAULT.
466
467
468
469
470
471

472 SET DTE TEST

473 PURPOSE:

474 IN ADDITION TO VERIFYING THAT 'DRIVE TIMING ERROR' CAN
475 BE SET BY THE CS MODULE, THIS TEST ALSO VERIFIES

476 * THAT 'MAINTENANCE SECTOR COMPARE' IS NOT STUCK AT ONE
477 OR ZERO.

478 * THAT 'ENABLE SEARCH' IS NOT STUCK AT ONE OR ZERO.
479

480 PROCEDURE:

481 (1) INITIALIZE AND VERIFY THAT 'DTE' IS RESET, THEN
482 SET MAINTENANCE INDEX PULSE AND VERIFY THAT DTE IS STILL
483 RESET. THIS TEST WILL INSURE THAT THE SECTOR COMPARE FLOP
484 IS NOT STUCK AT ONE.
485

486 (2) EXECUTE A DATA COMMAND IN MAINTENANCE MODE TO
487 THE POINT WHERE SEARCH IS ENABLED, I.E., P EN SEARCH H IS
488 IS ACTIVE. SET AND RESET THE SECTOR PULSE TO SET 'CS3 EN
489 SEARCH' FLOP, AND CLOCK 'CSS SECTOR COMPARE' FLOP WHICH
490 SHOULD NOT SET. SET SECTOR PULSE AND VERIFY THAT DTE IS
491 SET. THIS TEST FAILS IF J INPUT TO SECTOR COMPARE FLOP
492 IS STUCK AT ONE.
493

494 REPEAT, BUT SET MAINTENANCE SECTOR COMPARE AND VERIFY
495 THAT DTE SETS. THIS TEST FAILS IF MAINTENANCE SECTOR
496 COMPARE IS STUCK AT ZERO.
497

498 PROBABLE FAULT:

499 1. CS MODULE

500 2. DS MODULE
501
502
503
504
505
506
507
508
509
510
511
512
513

FORMAT CHANGE TEST

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

PURPOSE:

TO VERIFY THAT A CHANGE IN FORMAT INHIBITS SEARCH
ENABLE UNTIL THE NEXT INDEX PULSE.

PROCEDURE:

THE TEST WILL USE 'DTE' FOR VISIBILITY OF 'CS3 EN
SEARCH H'.

THE FOLLOWING STEPS ARE EXECUTED:

(1) INITIALIZE AND SET THE FORMAT TO 18 BIT MODE
TO SET 'CS3 FMT CHANGE' FLOP.

(2) EXECUTE A DATA COMMAND TO THE POINT WHERE SEARCH
IS ENABLED BY THE SEQUENCER.

(3) SET 'MAINTENANCE SECTOR COMPARE', THEN SET
'MAINTENANCE SECTOR PULSE' TO CLOCK 'CS3 EN SEARCH' FLOP
WHICH SHOULD NOT SET BECAUSE OF THE FORMAT CHANGE.

(4) RESET SECTOR PULSE TO CLOCK 'CS5 SECTOR COMPARE'
FLOP WHICH WILL NOT SET IF 'CS3 EN SEARCH H' IS INACTIVE.

(5) SET SECTOR PULSE AND VERIFY THAT DRIVE TIMING
ERROR IS RESET.

(6) SET AND RESET INDEX PULSE TO CLEAR THE FORMAT
CHANGE FLOP.

(7) SET AND RESET SECTOR PULSE TO SET 'CS3 EN SEARCH'
FLOP AND 'CS5 SECTOR COMPARE' FLOP.

(8) SET SECTOR COMPARE AND VERIFY THAT DTE IS SET.

REPEAT THE TEST WITH A FORMAT CHANGE FROM 18 BIT MODE
TO 16 BIT MODE.

PROBABLE FAULT:

1. CS MODULE

PROM STROBE TEST

PURPOSE:

TO VERIFY THAT WORD CLOCK AND PROM STROBE CAN BE MANIPULATED
IN MAINTENANCE MODE.

PROCEDURE:

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

INITIALIZE AND SET 16 BIT MODE, THEN SEQUENCE THE MAINTENANCE CLOCK UNTIL PROM STROBE SETS. ISSUE -- MORE MAINTENANCE CLOCK PULSES AND VERIFY THAT PROM STROBE RESETS.

PROBABLE FAULT:

1. CS MODULE
2. DS MODULE

SYNC WORD COUNT INHIBIT TEST

PURPOSE:

TO VERIFY THE FOLLOWING DURING READ COMMAND:

- * THAT 'CS4 P LFS' (LOOKING FOR SYNC) GOES ACTIVE.
- * THAT 'LOOKING FOR SYNC' INHIBITS THE WORD COUNT

PROCEDURE:

A READ COMMAND IS SETUP AND EXECUTED TO THE POINT WHERE 'LOOKING FOR SYNC' SHOULD BE ACTIVE, WITH THE PROGRAM VERIFYING THE TRANSITION OF THE SIGNAL. THE PROGRAM THEN SUPPLIES A SERIES OF BIT CLOCKS AND VERIFIES THAT 'PROM STROBE' NEVER GOES ACTIVE.

PROBABLE FAULT:

1. CS MODULE IF LFS FAILT,
2. DS MODULE IF PROM STROBE FAILS.

SYNC DETECTION TEST

PURPOSE:

TO VERIFY THAT THE APPROPRIATE SYNC PATTERN IS RECOGNIZED.

PROCEDURE:

THE TEST EXECUTES A READ COMMAND IN MAINTENANCE MODE, USING BIT 10 OF THE MAINTENANCE REGISTER (RMMR1) TO SIMULATE THE SYNC BIT STREAM, AND USING PROM STROBE TO DETERMINE IF THE SYNC PATTERN HAS BEEN DETECTED.

THE SYNC PATTERN IS 00011001, WITH THE LEFT MOST

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

BIT REPRESENTING THE LAST BIT OF THE STREAM.

PROBABLE FAULT:

1. DS MODULE
2. CS MODULE

ABORT SYNC DETECTION TEST

PURPOSE:

TO VERIFY THAT 'WORD COUNT INHIBIT' IS RESET IF A 'DRIVE TIMING ERROR' OCCURS DURING SYNC DETECTION.

PROCEDURE:

A READ COMMAND IS INITIATED IN MAINTENANCE MODE. WHEN 'LOOKING FOR SYNC' GOES ACTIVE, THE TEST FORCES A DRIVE TIMING ERROR AND USES PROM STROBE TO VERIFY THAT 'WORD COUNT INHIBIT' HAS BEEN RESET.

PROBABLE FAULT:

1. DS MODULE

SYNC GENERATION TEST

PURPOSE:

TO VERIFY THAT THE APPROPRIATE SYNC PATTERN IS GENERATED DURING A FORMAT OPERATION.

PROCEDURE:

THE TEST EXECUTES A WRITE HEADER AND DATA COMMAND IN MAINTENANCE MODE, AND VERIFIES THAT THE CORRECT SYNC PATTERN IS GENERATED BY MONITORING WRITE DATA AT THE MAINTENANCE REGISTER (RMMR1).

PROBABLE FAULT:

1. DS MODULE

WRITE HEADER TEST

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

PURPOSE:

TO TEST THE OPERATION OF (1) THE DATA BUFFER AND SHIFT REGISTER AS WELL AS (2) THE ECC GENERATION DURING WRITE.

PROCEDURE:

A WRITE HEADER AND DATA COMMAND IS EXECUTED IN MAINTENANCE MODE. THE TEST VERIFIES HEADER WORDS ONE AND TWO AND THE CRC CHARACTER USING THE WRITE DATA BIT OF THE MAINTENANCE REGISTER.

AN RM02/3 USES CYLINDER 822., TRACK 4., SECTOR 31. AND 16 BIT FORMAT FOR THE TEST, WHICH CORRESPONSE TO THE FOLLOWING:

HEADER:

WORD 1 - 1101001100110110

WORD 2 - 0000010000011111

AN RM05 USES CYLINDER 822., TRACK 18., SECTOR 31. AND 16 BIT FORMAT FOR THIS TEST, WHICH CORRESPONSE TO THE FOLLOWING:

HEADER:

WORD 1 - 1101001100110110

WORD 2 - 0001001000011111

PROBABLE FAULT:

- 1. DS MODULE OR MASSBUS MODULE

HEADER COMPARE TEST

PURPOSE:

TO CHECK THE OPERATION OF (1) THE SHIFT REGISTER AND DATA BUFFER AS WELL AS THE (2) CRC GENERATOR DURING READ.

PROCEDURE:

THE TEST EXECUTES A READ HEADER AND DATA COMMAND IN MAINTENANCE MODE USING BIT 10 OF THE MAINTENANCE REGISTER (RMMR1) TO SIMULATE THE DATA BITS FOR THE FIRST AND SECOND HEADER WORDS AS WELL AS THE CRC CHARACTER. THE CRC CHARACTER IS IS FAULTED, AND THE TEST VERIFIES THAT A CRC ERROR IS DETECTED. ADDITIONALLY, THE TEST VERIFIES THAT HEADER WORDS ONE AND TWO ARE CORRECTLY TRANSFERD TO MEMORY. THE TEST USES THE SAME HEADER AS IN THE PREVIOUS TEST EXCEPT

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

THAT BIT 15 IS INVERTED.

PROBABLE FAULT:

1. DS OR IF MODULE IF CRC ERROR NOT DETECTED;
2. DS OR MASSBUS MODULE IF DATA INCORRECT

ECC GENERATION TEST

PURPOSE:

TO CHECK ECC OPERATION DURING WRITE.

PROCEDURE:

A WRITE DATA COMMAND IS EXECUTED IN MAINTENANCE MODE. ALL ONES DATA FIELD IS USED, AND THE TEST VERIFIES THE ECC CHARACTER VIA THE WRITE DATA BIT OF THE MAINTENANCE REGISTER. THE DATA FIELD IS NOT VERIFIED.

PROBABLE FAULT:

1. DS MODULE

ECC DETECTION TEST

PURPOSE:

TO CHECK THE ECC GENERATION DURING READ.

PROCEDURE.

THE TEST EXECUTES A READ DATA COMMAND IN MAINTENANCE MODE, AN ALL ONES DATA FIELD IS USED, HOWEVER THE LAST DATA WORD IS FAULTED.

THE TEST VERIFIES (1) THAT AN ECC ERROR IS DETECTED AND THAT (2) THE POSITION AND PATTERN REGISTERS ARE VALID.

PROBABLE FAULT:

1. DS MODULE

1
683
684

```

;PROGRAM VERSION #001
.TITLE CZRMQAO RM05/3/2 DSKLS TST 2
;*COPYRIGHT (C) 1980
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY MIKE LEAVITT
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C4), 1980.
    
```

685

```

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;* SWITCH USE
;* -----
;* 15 HALT ON ERROR
;* 14 LOOP ON TEST
;* 13 INHIBIT ERROR TYPEOUTS
;* 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
    
```

686

```

.SBTTL BASIC DEFINITIONS
    
```

687
688

```

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

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

;*GENERAL PURPOSE REGISTER DEFINITIONS
000000 R0 = %0 ;:GENERAL REGISTER
000001 R1 = %1 ;:GENERAL REGISTER
000002 R2 = %2 ;:GENERAL REGISTER
000003 R3 = %3 ;:GENERAL REGISTER
000004 R4 = %4 ;:GENERAL REGISTER
    
```

```
000005 R5 = %5 ;;GENERAL REGISTER
000006 R6 = %6 ;;GENERAL REGISTER
000007 R7 = %7 ;;GENERAL REGISTER
000006 SP = %6 ;;STACK POINTER
000007 PC = %7 ;;PROGRAM COUNTER
```

;*PRIORITY LEVEL DEFINITIONS

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

;'SWITCH REGISTER' SWITCH DEFINITIONS

```
100000 SW15 = 100000
040000 SW14 = 40000
020000 SW13 = 20000
010000 SW12 = 10000
004000 SW11 = 4000
002000 SW10 = 2000
001000 SW09 = 1000
000400 SW08 = 400
000200 SW07 = 200
000100 SW06 = 100
000040 SW05 = 40
000020 SW04 = 20
000010 SW03 = 10
000004 SW02 = 4
000002 SW01 = 2
000001 SW00 = 1
001000 SW9=SW09
000400 SW8=SW08
000200 SW7=SW07
000100 SW6=SW06
000040 SW5=SW05
000020 SW4=SW04
000010 SW3=SW03
000004 SW2=SW02
000002 SW1=SW01
000001 SW0=SW00
```

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

```
100000 BIT15 = 100000
040000 BIT14 = 40000
020000 BIT13 = 20000
010000 BIT12 = 10000
004000 BIT11 = 4000
002000 BIT10 = 2000
001000 BIT09 = 1000
000400 BIT08 = 400
000200 BIT07 = 200
000100 BIT06 = 100
000040 BIT05 = 40
000020 BIT04 = 20
```

```

000010 BIT03 = 10
000004 BIT02 = 4
000002 BIT01 = 2
000001 BIT00 = 1
001000 BIT9=BIT09
000400 BIT8=BIT08
000200 BIT7=BIT07
000100 BIT6=BIT06
000040 BIT5=BIT05
000020 BIT4=BIT04
000010 BIT3=BIT03
000004 BIT2=BIT02
000002 BIT1=BIT01
000001 BIT0=BIT00
    
```

```

;*BASIC 'CPU' TRAP VECTOR ADDRESSES
000004 ERRVEC = 4 ;:TIME OUT AND OTHER ERRORS
000010 RESVEC = 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
000014 TBITVEC = 14 ;:'T' BIT
000014 TRTVEC = 14 ;:TRACE TRAP
000014 BPTVEC = 14 ;:BREAKPOINT TRAP (BPT)
000020 IOTVEC = 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024 PWRVEC = 24 ;:POWER FAIL
000030 EMTVEC = 30 ;:EMULATOR TRAP (EMT) **ERROR**
000034 TRAPVEC = 34 ;:'TRAP' TRAP
000060 TKVEC = 60 ;:TTY KEYBOARD VECTOR
000064 TPVEC = 64 ;:TTY PRINTER VECTOR
000240 PIRQVEC = 240 ;:PROGRAM INTERRUPT REQUEST VECTOR
    
```

689
 690
 691
 692
 693
 694
 695
 696
 697
 698
 699
 700
 701
 702
 703
 704
 705
 706
 707
 708
 709
 710
 711
 712
 713
 714
 715
 716
 717

.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)

```

000000 NOP = 000000 ;:NOP COMMAND
000002 ILF02 = 000002 ;:ILLEGAL COMMAND
000004 SEEK = 000004 ;:SEEK COMMAND
000006 RECAL = 000006 ;:RECALIBRATE COMMAND
000010 DRVCLR = 000010 ;:DRIVE CLEAR COMMAND
000012 RELEASE = 000012 ;:RELEASE COMMAND
000014 OFFSET = 000014 ;:OFFSET COMMAND
000016 RTC = 000016 ;:RETURN TO CENTERLINE COMMAND
000020 RIP = 000020 ;:READ IN PRESET COMMAND
000022 PAKACK = 000022 ;:PACK ACKNOWLEDGE COMMAND
000024 PACACK = PAKACK
000024 ILF24 = 000024 ;:ILLEGAL COMMAND
000026 ILF26 = 000026 ;:ILLEGAL COMMAND
000030 SEARCH = 000030 ;:SEARCH COMMAND
    
```

720	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
721	000050	WCD = 000050	: WRITE CHECK DATA COMMAND
722	000052	WCH = 000052	: WRITE CHECK HEADER AND DATA
723	000054	ILF54 = 000054	: ILLEGAL COMMAND
724	000056	ILF56 = 000056	: ILLEGAL COMMAND
725	000060	WD = 000060	: WRITE DATA COMMAND
726	000062	WH = 000062	: WRITE HEADER AND DATA COMMAND
727	000064	ILF64 = 000064	: ILLEGAL COMMAND
728	000066	ILF66 = 000066	: ILLEGAL COMMAND
729	000070	RD = 000070	: READ DATA COMMAND
730	000072	RH = 000072	: READ HEADER AND DATA COMMAND
731	000074	ILF74 = 000074	: ILLEGAL COMMAND
732	000076	ILF76 = 000076	: ILLEGAL COMMAND
733			
734		: *RMDA DISK ADDRESS REGISTER	
735			
736		: TRACK ADDRESS DEFINITIONS	
737	010000	TA16 = BIT12	: TRACK ADDRESS 16.
738	004000	TA8 = BIT11	: TRACK ADDRESS 8.
739	002000	TA4 = BIT10	: TRACK ADDRESS 4
740	001000	TA2 = BIT09	: TRACK ADDRESS 2
741	000400	TA1 = BIT08	: TRACK ADDRESS 1
742			
743		: SECTOR ADDRESS DEFINITIONS	
744	000020	SA16 = BIT04	: SECTOR ADDRESS 16.
745	000010	SA8 = BIT03	: SECTOR ADDRESS 8.
746	000004	SA4 = BIT02	: SECTOR ADDRESS 4
747	000002	SA2 = BIT01	: SECTOR ADDRESS 2
748	000001	SA1 = BIT00	: SECTOR ADDRESS 1
749			
750		: TRACK & SECTOR MASKS	
751	177400	TADMSK = 177400	: TRACK ADDRESS MASK
752	000377	SADMSK = 000377	: SECTOR ADDRESS MASK
753			
754		: *RMDS DRIVE STATUS REGISTER	
755			
756	100000	ATA = BIT15	: ATTENTION ACTIVE
757	040000	ERR = BIT14	: COMPOSITE ERROR
758	020000	PIP = BIT13	: POSITIONING IN PROGRESS
759	010000	MOL = BIT12	: MEDIUM ON LINE
760	004000	WRL = BIT11	: WRITE LOCK
761	002000	LBT = BIT10	: LAST BLOCK TRANSFERRED
762	001000	PGM = BIT09	: PROGRAMMABLE
763	000400	DPR = BIT08	: DRIVE PRESENT
764	000200	DRY = BIT07	: DRIVE READY
765	000100	VV = BIT06	: VOLUME VALID
766	000001	OM = BIT00	: OFFSET MODE ACTIVE
767			
768		: *RMER1 ERROR REGISTER #1	
769			

```

770      100000      DCK      = BIT15      ;DATA CHECK ERROR
771      040000      UNS      = BIT14      ;DRIVE UNSAFE
772      020000      OPI      = BIT13      ;OPERATION INCOMPLETE
773      010000      DTE      = BIT12      ;DRIVE TIMING ERROR
774      004000      WLE      = BIT11      ;WRITE LOCK ERROR
775      002000      IAE      = BIT10      ;INVALID ADDRESS ERROR
776      001000      AOE      = BIT09      ;ADDRESS OVERFLOW ERROR
777      000400      HCRC     = BIT08      ;HEADER CRC ERROR
778      000200      HCE      = BIT07      ;HEADER COMPARE ERROR
779      000100      ECH      = BIT06      ;ECC 'HARD' ERROR
780      000040      WCF      = BIT05      ;WRITE CLOCK FAILURE
781      000020      FER      = BIT04      ;FORMAT ERROR
782      000010      PAR      = BIT03      ;PARITY ERROR
783      000004      RMR      = BIT02      ;REGISTER MODIFICATION REFUSED
784      000002      ILR      = BIT01      ;ILLEGAL REGISTER
785      000001      ILF      = BIT00      ;ILLEGAL FUNCTION
786
787      115760      NDTMSK   = DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
788      ;'NDTMSK' IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
789      ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS
790
791      ;*RMAS ATTENTION SUMMARY REGISTER
792
793      000377      ATMSK    = 377          ;MASK FOR ATTENTION BITS
794
795      ;*RMLA LOOK AHEAD REGISTER
796
797      002000      SC4      = BIT10      ;SECTOR COUNT = 16
798      001000      SC3      = BIT09      ;SECTOR COUNT = 8
799      000400      SC2      = BIT08      ;SECTOR COUNT = 4
800      000200      SC1      = BIT07      ;SECTOR COUNT = 2
801      000100      SC0      = BIT06      ;SECTOR COUNT = 1
802
803      003700      SCTMSK   = 003700     ;SECTOR COUNT MASK
804
805      ;*RMMR1 MAINTENANCE REGISTER #1
806
807      ;WRITE ONLY BITS
808      100000      DBCK     = BIT15      ;DEBUG CLOCK
809      040000      DBEN     = BIT14      ;DEBUG CLOCK ENABLE
810      020000      DEBL     = BIT13      ;DIAGNOSTIC END OF BLOCK
811      010000      DTO      = BIT12      ;DIAGNOSTIC TIMEOUT
812      004000      MCLK     = BIT11      ;MAINTENANCE CLOCK
813      002000      MRD      = BIT10      ;READ DATA
814      001000      MUR      = BIT09      ;UNIT READY
815      000400      MOC      = BIT08      ;ON CYLINDER
816      000200      MSER     = BIT07      ;SEEK ERROR
817      000100      MDF      = BIT06      ;DRIVE FAULT
818      000040      MS       = BIT05      ;SECTOR PULSE
819      000010      MWP      = BIT03      ;WRITE PROTECT
820      000004      MI       = BIT02      ;INDEX PULSE
821      000002      MSC      = BIT01      ;SECTOR COMPARE
822      000001      DMD      = BIT00      ;DIAGNOSTIC MODE
823
824      ;READ ONLY BITS
825      100000      OCC      = BIT15      ;OCCUPIED
826      040000      RG       = BIT14      ;RUN AND GO

```

827	020000	EBL	= BIT13	:END OF BLOCK
828	010000	REX	= BIT12	:EXCEPTION
829	004000	ESRC	= BIT11	:ENABLE SEARCH
830	002000	PLFS	= BIT10	:LOOKING FOR SYNC
831	001000	ECRC	= BIT09	:ENABLE CRC OUT
832	000400	PDA	= BIT08	:DATA AREA
833	000200	PHA	= BIT07	:HEADER AREA
834	000100	CONT	= BIT06	:CONTINUE
835	000040	WC	= BIT05	:WORD CLOCK
836	000020	EECC	= BIT04	:ENABLE ECC OUT
837	000010	MWD	= BIT03	:WRITE DATA BIT
838	000004	LS	= BIT02	:LAST SECTOR
839	000002	LST	= BIT01	:LAST SECTOR AND TRACK
840	000001	DMD	= BIT00	:DIAGNOSTIC MODE
841	051401	MR1AAA	= DMD!MUR!DBEN!MOC!DTO	
842				
843		:*RMDT	DRIVE TYPE REGISTER	
844				
845	100000	NSA	= BIT15	:NOT SECTOR ADDRESSED = 0
846	040000	TAP	= BIT14	:TAPE DRIVE = 0
847	020000	MOH	= BIT13	:MOVING HEAD = 1
848	004000	DRQ	= BIT11	:DRIVE REQUEST REQUIRED
849				
850	020024	SNGPRT	= 020024	:SINGLE PORT DRIVE TYPE
851	024024	DULPRT	= 024024	:DUAL PORT DRIVE TYPE
852				
853		:*RMOF	OFFSET REGISTER	
854				
855	010000	FMT16	= BIT12	:16 BIT WORD FORMAT
856	004000	ECI	= BIT11	:ECC INHIBIT
857	002000	HCI	= BIT10	:HEADER COMPARE INHIBIT
858	000200	OFD	= BIT07	:OFFSET FORWARD
859	161577	XNUOF	= 161577	:UNSED BITS OF RMOF
860				
861		:*RMDC	DESIRED CYLINDER ADDRESS REGISTER	
862				
863	001777	CYLMSK	= 001777	:MASK FOR CYLINDER ADDRESS
864	176000	XNUDC	= 176000	:UNSED BITS OF RMDC
865				
866		:*RMMR2	MAINTENANCE REGISTER #2	
867				
868		:READ ONLY BITS		
869	100000	RQA	= BIT15	:PORT A REQUEST
870	040000	RQB	= BIT14	:PORT B REQUEST
871	020000	TAG	= BIT13	:TAG CONTROL
872	010000	TST	= BIT12	:COMMAND SEQUENCE TEST BIT
873	004000	CC	= BIT11	:CONTROL OR CYLINDER TAG
874	002000	CH	= BIT10	:CONTROL OR HEAD TAG
877	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

```

878      000001      BB00      = BIT00      ;TAG BUS
879
880      ;*RMER2 ERROR REGISTER 2
881      100000      BSE       = BIT15      ;BAD SECTOR ERROR
882      040000      SKI       = BIT14      ;SEEK INCOMPLETE
883      020000      OPE       = BIT13      ;OPERATOR PLUG ERROR
884      010000      IVC       = BIT12      ;INVALID COMMAND ERROR
885      004000      LSC       = BIT11      ;LOSS OF SYSTEM CLOCK
886      002000      LBC       = BIT10      ;LOSS OF BIT CLOCK
887      000200      DVC       = BIT07      ;DEVICE CHECK
888      000010      DPE       = BIT03      ;DATA PARITY ERROR
889      001567      XNUER2    = 001567    ;UNUSED BITS OF RMER2
890
891      .SBTTL      PROGRAM MNEMONICS
892
893      100000      MSE       = BIT15      ;MANUFACTURING DETECTED SECTOR ERROR
894      040000      USE       = BIT14      ;USER DETECTED SECTOR ERROR
895
896      .SBTTL      RM REGISTER INDEX VALUES
897
898      000000      RMCS1     = 00      ;CONTROL STATUS REGISTER #1
899      000006      RMDA     = 06      ;DISK ADDRESS REGISTER
900      000012      RMDS     = 12      ;DRIVE STATUS REGISTER
901      000014      RMER1     = 14      ;ERROR REGISTER #1
902      000016      RMAS     = 16      ;ATTENTION SUMMARY REGISTER
903      000020      RMLA     = 20      ;LOOK AHEAD REGISTER
904      000024      RMMR1     = 24      ;MAINTENANCE REGISTER
905      000026      RMDT     = 26      ;DRIVE TYPE REGISTER
906      000030      RMSN     = 30      ;SERIAL NUMBER REGISTER
907      000032      RMOF     = 32      ;OFFSET REGISTER
908      000034      RMDC     = 34      ;DESIRED CYLINDER REGISTER
909      000036      RMHR     = 36      ;HOLDING REGISTER
910      000040      RMMR2     = 40      ;MAINTENANCE REGISTER #2
911      000042      RMER2     = 42      ;ERROR REGISTER #2
912      000044      RMEC1     = 44      ;ECC POSITION REGISTER
913      000046      RMEC2     = 46      ;ECC PATTERN REGISTER
916      000050      ILRG50    = 50      ;ILLEGAL REGISTER 50
          000052      ILRG52    = 52      ;ILLEGAL REGISTER 52
          000054      ILRG54    = 54      ;ILLEGAL REGISTER 54
          000056      ILRG56    = 56      ;ILLEGAL REGISTER 56
          000060      ILRG60    = 60      ;ILLEGAL REGISTER 60
          000062      ILRG62    = 62      ;ILLEGAL REGISTER 62
          000064      ILRG64    = 64      ;ILLEGAL REGISTER 64
          000066      ILRG66    = 66      ;ILLEGAL REGISTER 66
          000070      ILRG70    = 70      ;ILLEGAL REGISTER 70
          000072      ILRG72    = 72      ;ILLEGAL REGISTER 72
          000074      ILRG74    = 74      ;ILLEGAL REGISTER 74
          000076      ILRG76    = 76      ;ILLEGAL REGISTER 76
917
918
919      000077      IDXMSK    = 77      ;MASK FOR REGISTER INDEX NUMBER
920
921      .SBTTL      RH CONTROLLER REGISTER BIT DEFINITIONS
922
923      ;*RMCS1 CONTROL STATUS REGISTER #1
924

```


925	100000	SC	= BIT15	:SPECIAL CONDITION-READ ONLY
926	040000	TRE	= BIT14	:TRANSFER ERROR
927	020000	MCPE	= BIT13	:MASSBUS CONTROL BUS PARITY ERROR-READ ONLY
928	002000	PSEL	= BIT10	:PORT B SELECT
929	001000	A17	= BIT09	:ADDRESS EXTENSION
930	000400	A16	= BIT08	:ADDRESS EXTENSION
931	000200	RDY	= BIT07	:READY-READ ONLY
932	000100	IE	= BIT06	:INTERRUPT ENABLE
933				
934				:*RMCS2 RH CONTROL STATUS REGISTER #2
935				
936	100000	DLT	= BIT15	:DATA LATE-READ ONLY
937	040000	WCE	= BIT14	:WRITE CHECK ERROR-READ ONLY
938	020000	UPE	= BIT13	:UNIBUS PARITY ERROR
939	010000	NED	= BIT12	:NONEXISTANT DRIVE-READ ONLY
940	004000	NEM	= BIT11	:NONEXISTANT MEMORY-READ ONLY
941	002000	PGE	= BIT10	:PROGRAM ERROR-READ ONLY
942	001000	MXF	= BIT09	:MISSED TRANSFER
943	000400	MDPE	= BIT08	:MASSBUS DATA BUS PARITY ERROR-READ ONLY
944	000200	OR	= BIT07	:OUTPUT READY-READ ONLY
945	000100	IR	= BIT06	:INPUT READY-READ ONLY
946	000040	CLR	= BIT05	:CONTROLLER CLEAR
947	000020	PAT	= BIT04	:PARITY TEST
948	000010	BAI	= BIT03	:UNIBUS ADDRESS INCREMENT INHIBIT
951	000004	U2	= BIT02	:UNIT SELECT
	000002	U1	= BIT01	:UNIT SELECT
	000001	U0	= BIT00	:UNIT SELECT
952				
953				:UNIT SELECT MASK
954				
955	000007	UNTMSK	= 7	:UNIT SELECT MASK
956				
957				:*RMCS3 RH70 CONTROL STATUS REGISTER #3
958				
959	100000	APE	= BIT15	:ADDRESS PARITY ERROR
960	040000	DPEHI	= BIT14	:DATA PARITY ERROR HIGH WORD
961	020000	DPELO	= BIT13	:DATA PARITY ERROR LOW WORD
962	010000	WCEHI	= BIT12	:WRITE CHECK ERROR HIGH WORD
963	004000	WCELO	= BIT11	:WRITE CHECK ERROR LOW WORD
964	002000	DBL	= BIT10	:DOUBLE WORD TRANSFER
965	000100	IE	= BIT06	:INTERRUPT ENABLE
966	000010	IPCK3	= BIT03	:INVERT PARITY CHECK
967	000004	IPCK2	= BIT02	:INVERT PARITY CHECK
968	000002	IPCK1	= BIT01	:INVERT PARITY CHECK
969	000001	IPCK0	= BIT00	:INVERT PARITY CHECK
970				
971				.SBTTL RH11/RH70 CONTROLLER REGISTER INDEX VALUES
972				
973	000000	RMCS1	= 00	:CONTROL, STATUS REGISTER #1
974	000002	RMWC	= 02	:WORD COUNT REGISTER
975	000004	RMBA	= 04	:BUS ADDRESS REGISTER
976	000010	RMCS2	= 10	:CONTROL, STATUS REGISTER #2
977	000022	RMDB	= 22	:DATA BUFFER
978	000050	RMBAE	= 50	:BUS ADDRESS EXTENSION
979	000052	RMCS3	= 52	:CONTROL, STATUS REGISTER #3
980				
981	176700	ABASE	= 176700	:UNIBUS ADDRESS

```

982      120254      AVECT1 = 120254      ;UNIBUS VECTOR ADDRESS AND PRIORITY
983
985
986      .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      ;;SOFTWARE DISPLAY REGISTER
000176    000000      SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER

          .SBTTL STARTING ADDRESS(ES)
987    000200    000137    002716      JMP      @#START      ;;JUMP TO STARTING ADDRESS OF PROGRAM
988
          .SBTTL ACT11 HOOKS
          ;*****
          ;HOOKS REQUIRED BY ACT11
          000204      $SVPC=.      ;SAVE PC
          000046      000046      .=46
          000046      021546      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
          000052      000052      .=52
          000052      000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
          000204      000204      .= $SVPC      ;; RESTORE PC

989
990      001100      .=1100
991      .SBTTL APT PARAMETER BLOCK
          ;*****
          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
          ;*****
          001100      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      000001      $TSTM:  .WORD 1      ;;RUN TIM OF LONGEST TEST
          001106      000002      $PASTM: .WORD 2      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
          001110      000002      $UNITM: .WORD 2      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
          001112      000042      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
992      001114      TAGADR=.
  
```


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	\$DEVN:	.WORD	ADEVN	::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			

```
0 .SBTTL USER DEFINED TAGS
001326 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.
001330 000 LSTRK: .BYTE 0 ;LO BYTE = 0
001331 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
```

001332

GETBUF:

```
;REGISTER INPUT BUFFER
001332 000000 RMCS1I: .WORD 0 ;CONTROL, STATUS REGISTER #1
001334 000000 RMWCI: .WORD 0 ;WORD COUNT REGISTER
001336 000000 RMBAI: .WORD 0 ;BUS ADDRESS REGISTER
001340 000000 RMDAI: .WORD 0 ;DISK ADDRESS REGISTER
001342 000000 RMCS2I: .WORD 0 ;CONTROL, STATUS REGISTER #2
001344 000000 RMDSI: .WORD 0 ;DRIVE STATUS REGISTER
001346 000000 RMER1I: .WORD 0 ;ERROR REGISTER #1
001350 000000 RMAI: .WORD 0 ;ATTENTION SUMMARY REGISTER
001352 000000 RMLAI: .WORD 0 ;LOOK AHEAD REGISTER
001354 000000 RMDBI: .WORD 0 ;DATA BUFFER
001356 000000 RMMR1I: .WORD 0 ;MAINTENANCE REGISTER #1
001360 000000 RMDTI: .WORD 0 ;DRIVE TYPE REGISTER
001362 000000 RMSNI: .WORD 0 ;SERIAL NUMBER REGISTER
001364 000000 RMOFI: .WORD 0 ;OFFSET REGISTER
001366 000000 RMDCI: .WORD 0 ;DESIRED CYLINDER REGISTER
001370 000000 RMHRI: .WORD 0 ;HOLDING REGISTER
001372 000000 RMMR2I: .WORD 0 ;MAINTENANCE REGISTER #2
001374 000000 RMER2I: .WORD 0 ;ERROR REGISTER #2
001376 000000 RMEC1I: .WORD 0 ;ECC POSITION REGISTER
001400 000000 RMEC2I: .WORD 0 ;ECC PATTERN REGISTER
001402 000000 RMBAEI: .WORD 0 ;BUS ADDRESS EXTENSION REGISTER
001404 000000 RMCS3I: .WORD 0 ;CONTROL, STATUS REGISTER #3
```

```
;THE REGISTER OUTPUT BUFFER IS USED FOR
;ASSEMBLING DATA GOING TO REGISTER
```

001406

PUTBUF:

```
;REGISTER OUTPUT BUFFER
001406 000000 RMCS1O: .WORD 0 ;CONTROL, STATUS REGISTER #1
001410 000000 RMWCO: .WORD 0 ;WORD COUNT REGISTER
001412 000000 RMBAO: .WORD 0 ;BUS ADDRESS REGISTER
001414 000000 RMDAO: .WORD 0 ;DISK ADDRESS REGISTER
001416 000000 RMCS2O: .WORD 0 ;CONTROL, STATUS REGISTER #2
001420 000000 RMDSO: .WORD 0 ;DRIVE STATUS REGISTER
001422 000000 RMER1O: .WORD 0 ;ERROR REGISTER #1
001424 000000 RMAO: .WORD 0 ;ATTENTION SUMMARY REGISTER
001426 000000 RMLAO: .WORD 0 ;LOOK AHEAD REGISTER
001430 000000 RMDBO: .WORD 0 ;DATA BUFFER
001432 000000 RMMR1O: .WORD 0 ;MAINTENANCE REGISTER #1
001434 000000 RMDTO: .WORD 0 ;DRIVE TYPE REGISTER
001436 000000 RMSNO: .WORD 0 ;SERIAL NUMBER REGISTER
```

001440	000000	RMOFO: .WORD	0	:OFFSET REGISTER
001442	000000	RMDCO: .WORD	0	:DESIRED CYLINDER REGISTER
001444	000000	RMHRO: .WORD	0	:HOLDING REGISTER
001446	000000	RMMR20: .WORD	0	:MAINTENANCE REGISTER #2
001450	000000	RMER20: .WORD	0	:ERROR REGISTER #2
001452	000000	RMEC10: .WORD	0	:ECC POSITION REGISTER
001454	000000	RMEC20: .WORD	0	:ECC PATTERN REGISTER
001456	000000	RMBAE0: .WORD	0	:BUS ADDRESS EXTENSION REGISTER
001460	000000	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.

001462	000000	TSTQUE: .WORD	0	:CONTAINS DEVICE POINTER
		.BLKW	8.	:TEST QUE FOR DEVICES UNDER TEST
001504	000000	.WORD	0	:TABLE TERMINATOR GOES HERE WHEN :A' 8. DEVICES ARE UNDER TEST.

001506	172540	\$LPCSR: .WORD	172540	:KW11-P CONTROL + STATUS REGISTER
001510	172542	\$LPCSB: .WORD	172542	:KW11-P COUNT SET BUFFER
001512	000104	\$LPVEC: .WORD	104	:KW11-P INTERRUPT VECTOR
001514	000106	.WORD	106	
001516	177546	\$LLCSR: .WORD	177546	:KW11-L CONTROL + STATUS REGISTER
001520	000100	\$LLVEC: .WORD	100	:KW11-L INTERRUPT VECTOR
001522	000102	.WORD	102	
001524	000000	\$PSW: .WORD		:STORAGE FOR PRIORITY
001526	000000	TIME: .WORD		:STORAGE FOR ELAPSED TIME
001530	000000	WATCH: .WORD		:STORAGE FOR REMAINING TIME
001532	000000	CLOCK: .WORD		:ADDRESS OF START CLOCK SUB
001534	000000	STOP: .WORD		:ADDRESS OF STOP CLOCK SUB

:PUT TAGS HERE

0

.SBTTL ERROR POINTER TABLE

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

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

001536
 1
 2
 3

\$ERRTB:
 ;ERROR 1 CANNOT CLEAR NED STATUS

001536 032024
 001540 037730
 001542 040044
 001544 040100

EMT1
 EHT1
 EDT1
 EFT1

4
 5
 6

;ERROR 2 CANNOT READ OR WRITE ANY DEVICE REG WITHOU: NED

001546 032032
 001550 037734
 001552 040046
 001554 040102

EMT2
 EHT2
 EDT2
 EFT2

7
 8
 9

;ERROR 3 CANNOT WRITE/READ ONES TO ANY DEVICE REGISTER

001556 032060
 001560 000000
 001562 000000
 001564 000000

EMT3
 0
 0
 0

10
 11
 12

;ERROR 4 CANNOT CLEAR ANY DEVICE REGISTER BITS W/MASSBUS INIT

001566 032100
 001570 000000
 001572 000000
 001574 000000

EMT4
 0
 0
 0

13
 14
 15

;ERROR 5 CANNOT WRITE/READ ZEROS TO ALL BIT POSITIONS

001576 032122
 001600 037740
 001602 040050
 001604 040104

EMT5
 EHT5
 EDT5
 EFT5

16
 17

;ERROR 6 CANNOT WRITE/READ ONES TO ALL BIT POSITIONS

18

001606	032146	EMT6
001610	037740	EHT5
001612	040050	EDT5
001614	040104	EFT5

19

20 :ERROR 7 CANNOT WRITE/READ SHIFTING ONE BIT TO ALL BIT POSITIONS
21 : OF DEVICE REGISTERS
22 :

001616	032170	EMT7
001620	037744	EHT7
001622	040050	EDT5
001624	040104	EFT5

23

24 :ERROR 10 REGISTER SELECT 1 APPEARS S-A-0
25 :

001626	032212	EMT10
001630	000000	0
001632	000000	0
001634	000000	0

26

27 :ERROR 11 REGISTER SELECT 1 APPEARS S-A-1
28 :

001636	032230	EMT11
001640	000000	0
001642	000000	0
001644	000000	0

29

30 :ERROR 12 REGISTER SELECT 2 APPEARS S-A-0
31 :

001646	032246	EMT12
001650	000000	0
001652	000000	0
001654	000000	0

32

33 :ERROR 13 REGISTER SELECT 2 APPEARS S-A-1
34 :

001656	032264	EMT13
001660	000000	0
001662	000000	0
001664	000000	0

35

36 :ERROR 14 REGISTER SELECT 4 APPEARS S-A-0
37 :

001666	032302	EMT14
001670	000000	0
001672	000000	0
001674	000000	0

38

39 :ERROR 15 REGISTER SELECT 4 APPEARS S-A-1

ERROR POINTER TABLE

40	001676	032320	EMT15	
	001700	000000	0	
	001702	000000	0	
	001704	000000	0	
41				
42			:ERROR 16	REGISTER SELECT 8 APPEARS S-A-0
43	001706	032336	EMT16	
	001710	000000	0	
	001712	000000	0	
	001714	000000	0	
44				
45			:ERROR 17	REGISTER SELECT 8 APPEARS S-A-1
46	001716	032354	EMT17	
	001720	000000	0	
	001722	000000	0	
	001724	000000	0	
47				
48			:ERROR 20	OPI SET WITH SEACH TIMEOUT DISABLED
49	001726	032372	EMT20	
	001730	037730	EHT1	
	001732	040044	EDT1	
	001734	040100	EFT1	
50				
51			:ERROR 21	OPI SET WITH SECTOR PULSE,SECTOR COMPARE WAS RESET
52	001736	032414	EMT21	
	001740	037730	EHT1	
	001742	040044	EDT1	
	001744	040100	EFT1	
53				
54			:ERROR 22	DTE SET WITH INDEX PULSE,SEARCH WAS NOT ENABLED
55	001746	032436	EMT22	
	001750	037730	EHT1	
	001752	040044	EDT1	
	001754	040100	EFT1	
56				
57			:ERROR 23	DTE SET WITH SECTOR PULSE,SECTOR COMPARE WAS RESET
58	001756	032462	EMT23	
	001760	037730	EHT1	
	001762	040044	EDT1	
	001764	040100	EFT1	
59				
60			:ERROR 24	DTE DID NOT SET WITH SECTOR PULSE,SECTOR COMPARE WAS SET
61				

	001766	032506	EMT24	
	001770	037730	EHT1	
	001772	040044	EDT1	
	001774	040100	EFT1	
62				
63				
64				:ERROR 25 DTE SET WITH SECTOR PULSE DURING FORMAT CHANGE
	001776	032532	EMT25	
	002000	037730	EHT1	
	002002	040044	EDT1	
	002004	040100	EFT1	
65				
66				
67				:ERROR 26 MBA CLR L IS STUCK ACTIVE
	002006	032552	EMT26	
	002010	000000	0	
	002012	000000	0	
	002014	000000	0	
68				
69				
70				:ERROR 27 COULD NOT SET DTE AFTER FORMAT CHANGE
	002016	032604	EMT27	
	002020	037730	EHT1	
	002022	040044	EDT1	
	002024	040100	EFT1	
71				
72				
73				:ERROR 30 CANNOT SET PROM STROBE WITH BIT CLOCK
	002026	032624	EMT30	
	002030	037730	EHT1	
	002032	040044	EDT1	
	002034	040100	EFT1	
74				
75				
76				:ERROR 31 CANNOT CLEAR RMER1,DTE
	002036	032644	EMT31	
	002040	037730	EHT1	
	002042	040044	EDT1	
	002044	040100	EFT1	
77				
78				
79				:ERROR 32 PROM STROBE RESET EARLY
	002046	032660	EMT32	
	002050	037730	EHT1	
	002052	040044	EDT1	
	002054	040100	EFT1	
80				
81				
82				:ERROR 33 PROM STROBE SET EARLY
	002056	032672	EMT33	

002060	037730	EHT1	
002062	040044	EDT1	
002064	040100	EFT1	
83			
84		:ERROR 34	LOOKING FOR SYNC SET EARLY
85			
002066	032704	EMT34	
002070	037730	EHT1	
002072	040044	EDT1	
002074	040100	EFT1	
86			
87		:ERROR 35	LOOKING FOR SYNC DID NOT SET
88			
002076	032716	EMT35	
002100	037730	EHT1	
002102	040044	EDT1	
002104	040100	EFT1	
89			
90		:ERROR 36	PROM STROBE SET WHILE LOOKING FOR SYNC
91			
002106	032730	EMT36	
002110	037730	EHT1	
002112	040044	EDT1	
002114	040100	EFT1	
92			
93		:ERROR 37	SYNC DETECTED WITH WRONG PATTERN
94			
002116	032750	EMT37	
002120	040004	EHT115	
002122	040070	EDT115	
002124	040116	EFT115	
95			
96		:ERROR 40	SYNC NOT DETECTED
97			
002126	033000	EMT40	
002130	037730	EHT1	
002132	040044	EDT1	
002134	040100	EFT1	
98			
99		:ERROR 41	DRIVE TIMING ERROR DID NOT CLEAR LOOKING FOR SYNC
100			
002136	033022	EMT41	
002140	037730	EHT1	
002142	040044	EDT1	
002144	040100	EFT1	
101			
102		:ERROR 42	WRITE GATE DID NOT COME ON OR RESET EARLY
103			
002146	033050	EMT42	
002150	037730	EHT1	

	002152	040044	EDT1	
	002154	040100	EFT1	
104				
105			:ERROR 43	INCORRECT SYNC PATTERN DURING HEADER
106				
	002156	033066	EMT43	
	002160	000000	0	
	002162	000000	0	
	002164	000000	0	
107				
108			:ERROR 44	INCORRECT SYNC PATTERN DURING HEADER
109				
	002166	033066	EMT43	
	002170	037744	EHT7	
	002172	040050	EDT5	
	002174	040104	EFT5	
110				
111			:ERROR 45	HEADER AREA DID NOT COME ON OR RESET EARLY
112				
	002176	033120	EMT45	
	002200	037730	EHT1	
	002202	040044	EDT1	
	002204	040100	EFT1	
113				
114			:ERROR 46	CRC ENABLE DID NOT SET
115				
	002206	033134	EMT46	
	002210	037730	EHT1	
	002212	040044	EDT1	
	002214	040100	EFT1	
116				
117			:ERROR 47	INCORRECT HEADER GENERATED DURING WRITE
118				
	002216	033150	EMT47	
	002220	037750	EHT47	
	002222	040052	EDT47	
	002224	040100	EFT1	
119				
120			:ERROR 50	READ GATE INCORRECT DURING HEADER AREA
121				
	002226	033166	EMT50	
	002230	037730	EHT1	
	002232	040044	EDT1	
	002234	040100	EFT1	
122				
123			:ERROR 51	UNEXPECTED HEADER ERROR DURING DIAGNOSTIC MODE
124				
	002236	033204	EMT51	
	002240	037730	EHT1	
	002242	040044	EDT1	

	002244	040100	EFT1	
125				
126			:ERROR 52	INCORRECT HEADER READ IN DIAGNOSTIC MODE
127				
	002246	033220	EMT52	
	002250	037754	EHT52	
	002252	040054	EDT52	
	002254	040106	EFT57	
128				
129			:ERROR 53	INCORRECT TAG BUS DURING DATA COMMAND
130				
	002256	037624	EMT276	
	002260	037730	EHT1	
	002262	040044	EDT1	
	002264	040100	EFT1	
131				
132			:ERROR 54	DATA TIMING SEQUENCER CONTROLS INCORRECT DURING DATA COMMAND
133				
	002266	033252	EMT54	
	002270	037730	EHT1	
	002272	040044	EDT1	
	002274	040100	EFT1	
134				
135			:ERROR 55	DATA AREA DID NOT COME ON OR RESET EARLY
136				
	002276	033270	EMT55	
	002300	037730	FHT1	
	002302	040044	EDT1	
	002304	040100	EFT1	
137				
138			:ERROR 56	ECC ENABLE DID NOT SET
139				
	002306	033306	EMT56	
	002310	037730	EHT1	
	002312	040044	EDT1	
	002314	040100	EFT1	
140				
141			:ERROR 57	DEVICE IS NOT AN RM05/3/2
142				
	002316	033322	EMT57	
	002320	037760	EHT57	
	002322	040056	EDT57	
	002324	040106	EFT57	
143				
144			:ERROR 60	DEVICE AVAILABLE IS NOT SET
145				
	002326	033336	EMT60	
	002330	037730	EHT1	
	002332	040044	EDT1	
	002334	040100	EFT1	

146				
147			:ERROR 61	INCORRECT ECC PATTERN GENERATED DURING WRITE
148				
	002336	033352	EMT61	
	002340	037764	EHT61	
	002342	040060	EDT61	
	002344	040106	EFT57	
149				
150			:ERROR 62	CANNOT CLEAR PROM STROBE WITH BIT CLOCK
151				
	002346	033370	EMT62	
	002350	037730	EHT1	
	002352	040044	EDT1	
	002354	040100	EFT1	
152				
153			:ERROR 63	DATA AREA DID NOT RESET
154				
	002356	033406	EMT63	
	002360	037730	EHT1	
	002362	040044	EDT1	
	002364	040100	EFT1	
155				
156			:ERROR 64	UNEXPECTED ECC ERROR IN DIAGNOSTIC MODE
157				
	002366	033420	EMT64	
	002370	037730	EHT1	
	002372	040044	EDT1	
	002374	040100	EFT1	
158				
159			:ERROR 65	INCORRECT DATA TRANSFERRED TO MEMORY
160				
	002376	033434	EMT65	
	002400	037730	EHT1	
	002402	040044	EDT1	
	002404	040100	EFT1	
161				
162			:ERROR 66	
163				
	002406	033446	EMT66	
	002410	000000	0	
	002412	000000	0	
	002414	000000	0	
164				
165			:ERROR 67	
166				
	002416	033462	EMT67	
	002420	000000	0	
	002422	000000	0	
	002424	000000	0	

167
168 :ERROR 70
169
002426 033500 EMT70
002430 000000 0
002432 000000 0
002434 000000 0

170
171 :ERROR 71
172
002436 033520 EMT71
002440 000000 0
002442 000000 0
002444 000000 0

173
174 :ERROR 72
175
002446 033544 EMT72
002450 000000 0
002452 000000 0
002454 000000 0

176
177 :ERROR 73
178
002456 033570 EMT73
002460 000000 0
002462 000000 0
002464 000000 0

179
180 :ERROR 74
181
002466 033610 EMT74
002470 000000 0
002472 000000 0
002474 000000 0

182
183 :ERROR 75
184
002476 033620 EMT75
002500 000000 0
002502 000000 0
002504 000000 0

185
186 :ERROR 76
187
002506 033632 EMT76
002510 000000 0
002512 000000 0
002514 000000 0

188

189				
190			;ERROR 77	
	002516	033646		EMT77
	002520	000000		0
	002522	000000		0
	002524	000000		0
191				
192			;ERROR 100	CANT SET VOLUME VALID
193				
	002526	035454		EMT170
	002530	040030		EHT150
	002532	040070		EDT115
	002534	040116		EFT115
194				
195			;ERROR 101	RUN AND GO WONT SET
196				
	002536	037604		EMT275
	002540	037730		EHT1
	002542	040044		EDT1
	002544	040100		EFT1
197				
198			;ERROR 102	SEARCH ENABLE WONT SET
199				
	002546	033722		EMT102
	002550	037730		EHT1
	002552	040044		EDT1
	002554	040100		EFT1
200				
201			;ERROR 103	OCCUPIED IS INCORRECT FOR FUNCTION CODE
202				
	002556	035520		EMT173
	002560	040030		EHT150
	002562	040070		EDT115
	002564	040116		EFT115
203				
204			;ERROR 104	READ IN PRESET DIDNT CLEAR RMDA, RMDC OR RMOF
205				
	002566	035542		EMT174
	002570	000000		0
	002572	000000		0
	002574	000000		0
206				
207			;ERROR 105	READ IN PRESET DIDNT CLEAR RMOF
208				
	002576	035570		EMT175
	002600	037730		EHT1
	002602	040044		EDT1
	002604	040100		EFT1
209				
210			;ERROR 106	READ IN PRESET DIDNT CLEAR RMDA

211 002606 035606 EMT176
 002610 037730 EHT1
 002612 040044 EDT1
 002614 040100 EFT1

212
213 ;ERROR 107 READ IN PRESET DIDNT CLEAR RMDC
214

 002616 035624 EMT177
 002620 037730 EHT1
 002622 040044 EDT1
 002624 040100 EFT1

215
216 ;ERROR 110 CANT SET OFFSET MODE BY OFFSET COMMAND
217

 002626 035642 EMT200
 002630 037730 EHT1
 002632 040044 EDT1
 002634 040100 EFT1

218
219 ;PUT ERROR TABLE HERE
220

```

1          ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS
2
3 002636 011600 BADTMO: MOV      (SP),R0          ;SAVE PC WHERE THE TIME OUT OCCURED
4 002640 005740      TST      -(R0)          ;ADJUST PC -2
5 002642 022626      CMP      (SP)+,(SP)+    ;RESTORE STACK POINTER
6 002644 104401 002652  TYPE     65$         ;:TYPE ASCIZ STRING
   002650 000417      BR       64$         ;:GET OVER THE ASCIZ
   ;:65$: .ASCIZ <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
   64$:
7 002710 010046      MOV      R0,-(SP)       ;SETUP FOR TYPING OUT PC
8 002712 104402      TYPOC
9 002714 000240      NOP                    ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
   ;TO STOP ON UNEXPECTED TIMEOUT.
10
11
12          .SBTTL  START OF PROGRAM
13
14 002716 000240      START:  NOP
15 002720 005227 000000  INC      #0          ;TTY LOOP, WAIT FOR INCREMENT
16 002724 001375      BNE     -4          ;OF WORD
17 002726 000005      RESET    ;RESET THE WORLD
18
19          .SBTTL  INITIALIZE THE COMMON TAGS
   ;:CLEAR THE COMMON TAGS ($CMTAG) AREA
   MOV      #SCMTAG,R6      ;:FIRST LOCATION TO BE CLEARED
   CLR      (R6)+          ;:CLEAR MEMORY LOCATION
   CMP      #SWR,R6        ;:DONE?
   BNE     -6             ;:LOOP BACK IF NO
   MOV      #STACK,SP     ;:SETUP THE STACK POINTER
   ;:INITIALIZE A FEW VECTORS
   MOV      #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
   MOV      #340,@IOTVEC+2 ;:LEVEL 7
   MOV      #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
   MOV      #340,@EMTVEC+2 ;:LEVEL 7
   MOV      #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
   MOV      #340,@TRAPVEC+2 ;:LEVEL 7
   MOV      #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      $ESCAPE       ;:CLEAR THE ESCAPE ON ERROR ADDRESS
   MOVB    #1,$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
   64$:  MOV      #65$, (SP) ;:SET UP FOR TRAP RETURN
   65$:  MOV      #SWREG,SWR ;:POINT TO SOFTWARE SWR
   MOV      #DISPREG,DISPLAY
    
```

```

003152 012637 000004      66$:  MOV      (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
003156 005037 001230      CLR      $PASS                ;;CLEAR PASS COUNT
003162 132737 000200 001243  BITB     #APTSIZE,$ENVM      ;;TEST USER SIZE UNDER APT
003170 001403      BEQ      67$                  ;;YES,USE NON-APT SWITCH
003172 012737 001244 001154  MOV      #$$SWREG,$SWR      ;;NO,USE APT SWITCH REGISTER
003200
20      67$:
21      ;SETUP 'TIMEOUT' TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
22      MOV      #BADTMO,ERRVEC  ;;SETUP FOR UNEXPECTED TIMEOUT
23      MOV      #PR6,ERRVEC+2   ;;LEVEL 6
003214 012746 000140      MOV      #PR3,-(SP)         ;;PUT NEW PS ON STACK
003220 012746 003226      MOV      #68$,-(SP)        ;;PUT NEW PC ON STACK
003224 000002      RTI                       ;;POP NEW PC AND PS
003226
24      68$:
25      .SBTTL  TYPE PROGRAM NAME
        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
003226 005227 177777      INC      #-1                ;;FIRST TIME?
003232 001055      BNE      69$                ;;BRANCH IF NO
003234 022737 021546 000042  CMP      #SENDAD,@#42       ;;ACT-11?
003242 001451      BEQ      69$                ;;BRANCH IF YES
003244 104401 003312      TYPE     ,70$               ;;TYPE ASCIZ STRING
        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
003250 005737 000042      TST      @#42               ;;ARE WE RUNNING UNDER XXDP/ACT?
003254 001012      BNE      71$                ;;BRANCH IF YES
003256 123727 001242 000001  CMPB     $ENV,#1            ;;ARE WE RUNNING UNDER APT?
003264 001406      BEQ      71$                ;;BRANCH IF YES
003266 023727 001154 000176  CMP      $SWR,#SWREG        ;;SOFTWARE SWITCH REG SELECTED?
003274 001005      BNE      72$                ;;BRANCH IF NO
003276 104407      GTSWR                      ;;GET SOFT-SWR SETTINGS
003300 000403      BR      72$
003302 112737 000001 001150 71$:  MOVB     #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
003310 72$:
003310 000426      BR      69$                ;;GET OVER THE ASCIZ
        ;;70$: .ASCIZ <CRLF>@CZRMQAO - RM05/3/2 DISKLESS TEST, PART 2@<CRLF>
        69$:
26
27      ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
28      ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
29
30      CLR      XXDP            ;;CLEAR 'XXDP' LOAD DEVICE STORAGE
31      CMPB     #16,@#41        ;;LOADED FROM AN RM05/3/2 ?
32      BNE      5$              ;;BRANCH IF NOT
33      MOV      @#40,XXDP       ;;GET DEVICE INDICATOR AND NUMBER
34      CMPB     #7,XXDP        ;;IS IT A VALID NUMBER ?
35      BHIS     1$              ;;YES
36      CLRB     XXDP           ;;NO, DEFAULT TO DRIVE 0
37      TST      @#42           ;;CHAIN MODE OR ACT11 AUTO ACCEPT ?
38      BEQ      3$              ;;BR IF NEITHER
39      TYPE     ,74$           ;;TYPE ASCIZ STRING
003436 000412      BR      73$               ;;GET OVER THE ASCIZ
        ;;74$: .ASCIZ <CRLF>/NOT TESTING DRIVE /
        73$:
40      CLR      -(SP)           ;;CLEAR WORD ON STACK
41      MOVB     XXDP,(SP)      ;;GET DRIVE ADDRESS
42      TYPOS    1              ;;TYPE THE ADDRESS
43      .BYTE    1              ;;ONLY 1 CHARACTER
    
```

```

44 003475      000          .BYTE 0          ;SUPRESS LEADING ZEROS
45 003476 104401 001217    TYPE  $CRLF      ;CR-LF
46 003502 000517          BR    5$          ;GET NUMBER OF DRIVES
47
48 003504 005227 177777    3$:  INC  #-1      ;FIRST TIME THRU HERE ?
49 003510 001114          BNE  5$          ;NO
50 003512 104401 003520    TYPE  76$       ;:TYPE ASCIZ STRING
    003516 000410          BR    75$       ;:GET OVER THE ASCIZ
    ;:76$: .ASCIZ <CRLF>/TO TEST DRIVE /
    75$:
51 003540 005046          CLR  -(SP)       ;CLEAR WORD ON STACK
52 003542 113716 001326    MOVB XXDP,(SP)  ;GET DRIVE ADDRESS
53 003546 104403          TYPOS          ;TYPE DRIVE ADDRESS
54 003550      001          .BYTE 1         ;ONLY 1 CHARACTER
55 003551      000          .BYTE 0         ;SUPRESS LEADING ZEROS
56 003552 104401 003560    TYPE  78$       ;:TYPE ASCIZ STRING
    003556 000431          BR    77$       ;:GET OVER THE ASCIZ
    ;:78$: .ASCIZ /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
    77$:
57 003642 104401 003650    TYPE  79$       ;:TYPE ASCIZ STRING
    003646 000435          BR    5$          ;:GET OVER THE ASCIZ
    ;:79$: .ASCIZ /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
    5$:
61 003742
62 003742 105737 001150    ;CHECK FOR AUTO MODE OR STANDALONE
63 003746 001506          TSTB $AUTOB     ;RUNNING IN AUTO MODE ?
64 003750 012737 000377 001300 BEQ  STANDALONE ;BR IF NO
65
66
67 003756          MOV  #377,$DEV  ;SET DEVICE MAP FOR ALL DRIVES
68 003756 132737 000200 001243 ;PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
69 003764 001066          XSIZ:
70
71 003766 005001          BITB #BIT7,$ENVM ;SIZING ALLOWED ?
72 003770 013700 001276    BNE  7$          ;NO
73
74 003774 136137 032014 001300 1$:  CLR  R1         ;START FROM DRIVE 0
75 004002 001453          MOV  $BASE,R0   ;LOAD THE BASE ADDRESS
76 004004 012737 031637 004130 BITB ATNTBL(R1),$DEV ;IS DEVICE PRESENT IN MAP ?
77 004012 005737 001326    BEQ  6$         ;BR IF NO
78 004016 001403          MOV  #LODEV,5$  ;GET ADDRESS OF LOAD DEVICE MESSAGE
79 004020 123701 001326    TST  XXDP       ;LOADED FROM RM05/3/2 ?
80 004024 001426          BEQ  2$         ;NO
81
82 004026 012760 000040 000010 2$:  MOV  #CLR, RMCS2(R0) ;CLEAR MASS BUS
83 004034 010160 000010          MOV  R1, RMCS2(R0) ;LOAD THE DRIVE ADDRESS
84 004040 005760 000012          TST  RMD5(R0)     ;TRY TO ACCESS AN RM DRIVE REGISTER
85 004044 012737 031657 004130 MOV  #NOTPRS,5$  ;GET ADDRESS OF NOT PRESENT MESSAGE
86 004052 032760 010000 000010 BIT  #NED, RMCS2(R0) ;IS DRIVE PRESENT ?
87 004060 001010          BNE  3$         ;BR IF NO
88 004062 012737 031674 004130 MOV  #NOTAVL,5$  ;GET ADDRESS OF AVAILABLE MESSAGE
89 004070 032760 004000 000000 BIT  #DVA, RMCS1(R0) ;IS DRIVE AVAILABLE ?
90 004076 001401          BEQ  3$         ;BR IF NO
91 004100 000414          BR    6$
92
93 004102 146137 032014 001300 3$:  BICB ATNTBL(R1),$DEV ;CLEAR DEVICE FROM BIT MAP
94 004110 104401 001217    4$:  TYPE  $CRLF      ;CR-LF
    
```

95	004114	104401	031631		TYPE	.MSGDRV	:TYPE 'DRIVE'
96	004120	010146			MOV	R1,-(SP)	::SAVE R1 FOR TYPEOUT
	004122	104403			TYPOS		::GO TYPE--OCTAL ASCII
	004124	002			.BYTE	2	::TYPE 2 DIGIT(S)
	004125	000			.BYTE	0	::SUPPRESS LEADING ZEROS
97	004126	104401			TYPE		:TYPE ERROR MESSAGE
98	004130	000000		5\$:	.WORD	0	:ADDRESS OF MESSAGE GOES HERE
99							
100	004132	005201		6\$:	INC	R1	:INCREMENT THE DRIVE ADDRESS
101	004134	020127	000007		CMP	R1,#7	:ALL DRIVES ARE CHECKED ?
102	004140	003715			BLE	1\$:BRANCH IF NOT
103							
104	004142	104401	001217	7\$:	TYPE	.\$CRLF	:CR-LF
105	004146	005004			CLR	R4	:THESE TWO LOOPS ARE ADDED TO
106	004150	005304			DEC	R4	:WAIT FOR TTY TO FINISH TYPING.
107	004152	001376			BNE	.-2	
108	004154	005304			DEC	R4	
109	004156	001376			BNE	.-2	
110							
111	004160	000137	005070		JMP	CMNSTART	:JUMP TO COMMON START
112							

57	004404	104401	031103		TYPE	,QUES		:TYPE " ? "
58	004410	104413			RDOCT			:GET NEW BUS ADDRESS
59	004412	012637	001176		MOV	(SP)+,\$TMP1		:CARRIAGE RETURN ?
60	004416	001412			BEQ	8\$:YES-SKIP TO NEXT ENTRY
61	004420	022737	160000	001176	CMP	#160000,\$TMP1		:BASE ADDRESS IN I/O PAGE ?
62	004426	101403			BLOS	7\$:YES
63	004430	104401	031247		TYPE	,CNSL02		:TYPE WARNING MESSAGE
64	004434	000760			BR	6\$+4		:TRY AGAIN
65	004436	013737	001176	001276	7\$:	MOV	\$TMP1,\$BASE	:STORE NEW BUS ADDRESS
66								
67	004444	104401	031311		8\$:	TYPE	,CNSL03	
68	004450	005046			CLR	-(SP)		
69	004452	113716	001272		MOVB	\$VECT1,(SP)		:GET CURRENT VECTOR ADDRESS
70	004456	104403			TYPOS			
71	004460	003			.BYTE	3		:TYPE 3 DIGITS
72	004461	000			.BYTE	0		:SUPPRESS LEADING ZEROS
73	004462	104401	031103		TYPE	,QUES		:TYPE " ? "
74	004466	104413			RDOCT			:GET NEW VECTOR ADDRESS
75	004470	012637	001176		MOV	(SP)+,\$TMP1		:CARRIAGE RETURN?
76	004474	001412			BEQ	10\$:YES-SKIP TO NEXT ENTRY
77	004476	022737	001000	001176	CMP	#1000,\$TMP1		:VECTOR ADDRESS < 1000 ?
78	004504	101003			BHI	9\$:YES!!
79	004506	104401	031331		TYPE	,CNSL04		:TYPE WARNING MESSAGE
80	004512	000754			BR	8\$: TRY
81	004514	113737	001176	001272	9\$:	MOVB	\$TMP1,\$VECT1	:STORE NEW VECTOR ADDRESS
82								
83	004522	104401	031365		10\$:	TYPE	,CNSL05	
84	004526	005046			CLR	-(SP)		
85	004530	113716	001273		MOVB	\$VECT1+1,(SP)		:GET CURRENT BR LEVEL
86	004534	006216			ASR	(SP)		
87	004536	006216			ASR	(SP)		
88	004540	006216			ASR	(SP)		
89	004542	006216			ASR	(SP)		
90	004544	006216			ASR	(SP)		
91	004546	104403			TYPOS			
92	004550	001			.BYTE	1		:ONLY 1 DIGIT
93	004551	000			.BYTE	0		:SUPPRESS LEADING ZEROS
94	004552	104401	031103		TYPE	,QUES		
95	004556	104413			RDOCT			:GET NEW PRIORITY
96	004560	012637	001176		MOV	(SP)+,\$TMP1		:CARRIAGE RETURN ?
97	004564	001424			BEQ	12\$:YES-SKIP TO NEXT ENTRY
98	004566	023727	001176	000007	CMP	\$TMP1,#7		:LEGAL PRIORITY ?
99	004574	002403			BLT	11\$:YES!!
100	004576	104401	031377		TYPE	,CNSL06		:TYPE WARNING MESSAGE
101	004602	000747			BR	10\$:TRY AGAIN
102	004604	006337	001176		11\$:	ASL	\$TMP1	:STORE NEW PRIORITY
103	004610	006337	001176		ASL	\$TMP1		
104	004614	006337	001176		ASL	\$TMP1		
105	004620	006337	001176		ASL	\$TMP1		
106	004624	006337	001176		ASL	\$TMP1		
107	004630	113737	001176	001273	MOVB	\$TMP1,\$VECT1+1		
108								
109								:DIALOGUE TO INPUT DEVICE NUMBERS
110	004636				12\$:	INC	#-1	:FIRST TIME THRU ?
111	004636	005227	177777		BNE	13\$:BR IF NO
112	004642	001002			TYPE	,CNSL07		:TYPE INPUT INSTRUCTIONS
113	004644	104401	031430					

```

114 004650 104401 001217      13$:  TYPE      ,SCLRF      ;CR-LF
115 004654 005037 001300      14$:  CLR        $DEVM    ;CLEAR DEVICE MAP
116 004660 104401 031615          TYPE      ,MSDRVS    ;TYPE 'DRIVE(S): '
117 004664 104411          RDCHR
118 004666 012637 001176          MOV      (SP)+,$TMP1  ;GET RESPONSE
119 004672 023727 001176 000101      CMP      $TMP1,#'A    ;IS INPUT 'A' ?
120 004700 001007          BNE      15$         ;NO
121 004702 104401 031076          TYPE      ,ALL       ;YES, TYPE 'ALL' AND GO
122 004706 012737 000377 001300      MOV      #377,$DEVM  ;SET DEVICE MAP FOR ALL DRIVES
123 004714 000137 003756          JMP      XSIZ       ;AUTO SIZE.
124
125 004720 023727 001176 000015 15$:  CMP      $TMP1,#CR   ;CARRIAGE RETURN ?
126 004726 001436          BEQ      17$         ;YES
127 004730 104401 001176          TYPE      , $TMP1    ;ECHO RESPONSE
128 004734 023727 001176 000060      CMP      $TMP1,#'0   ;NUMBER < 0 ?
129 004742 002430          BLT     17$         ;YES
130 004744 023727 001176 000067      CMP      $TMP1,#'7   ;NUMBER > 7 ?
131 004752 003427          BLE     18$         ;NO
132 004754 000423          BR      17$         ;ILLEGAL INPUT
133
134 004756 104411          16$:  RDCHR
135 004760 012637 001176          MOV      (SP)+,$TMP1  ;GET RESPONSE
136 004764 023727 001176 000015      CMP      $TMP1,#CR   ;CARRIAGE RETURN ?
137 004772 001432          BEQ     19$         ;YES
138 004774 104401 031107          TYPE      ,COMMA     ;TYPE ','
139 005000 104401 001176          TYPE      , $TMP1    ;ECHO RESPONSE
140 005004 023727 001176 000060      CMP      $TMP1,#'0   ;NUMBER < 0 ?
141 005012 002404          BLT     17$         ;YES
142 005014 023727 001176 000067      CMP      $TMP1,#'7   ;NUMBER > 7 ?
143 005022 003403          BLE     18$         ;NO
144 005024 104401 031573          17$:  TYPE      ,CNSLO8   ;TYPE '' ?ILLEGAL INPUT''
145 005030 000711          BR      14$         ;RETRY
146
147 005032 013701 001176          18$:  MOV      $TMP1,R1    ;R1 = DRIVE NUMBER
148 005036 042701 177770          BIC     #'C7,R1
149 005042 156137 032014 001300      BISB    ATNTBL(R1),$DEVM ;SET DEVICE IN MAP
150 005050 122737 000377 001300      CMPB   #377,$DEVM  ;DONE ?
151 005056 101337          BHI     16$         ;NO
152 005060 104401 001217          19$:  TYPE      ,SCLRF    ;CR-LF
153 005064 000137 003756          JMP     XSIZ       ;GO SIZE DEVICES
154

```



```

1          :ASSEMBLE TEST QUE FROM DEVICE MAP
2 005070   CMNSTART:
3 005070   013700 001300   MOV    $DEVN,R0           ;R0 = DEVICE MAP
4 005074   012701 001464   MOV    #TSTQUE+2,R1      ;R1 = ADDRESS OF FIRST ENTRY IN QUE
5 005100   010137 001462   MOV    R1,TSTQUE        ;INITIALIZE ENTRY POINTER
6 005104   012702 000001   MOV    #1,R2            ;R2 = DEVICE POINTER
7 005110   005003           CLR    R3                ;R3 = DEVICE NUMBER
8 005112   030200           1$:   BIT    R2,R0           ;IS THIS DEVICE IN MAP ?
9 005114   001406           BEQ    2$                ;NO !!
10 005116  010311           MOV    R3,(R1)          ;YES - ENTER DEVICE NUMBER IN QUE
11 005120  116361 032014 000001  MOVB  ATNTBL(R3),1(R1)  ;ENTER ATTENTION BIT IN QUE
12 005126  062701 000002           ADD    #2,R1            ;ADVANCE ENTRY POINTER
13 005132  006302           2$:   ASL    R2                ;ADVANCE DEVICE POINTER
14 005134  105702           TSTB  R2                ;DONE ALL DEVICES ?
15 005136  001402           BEQ    3$                ;YES
16 005140  005203           INC    R3                ;ADVANCE DEVICE NUMBER
17 005142  000763           BR    1$                ;ENTER NEXT DEVICE
18 005144  005011           3$:   CLR    (R1)          ;TERMINATE TEST QUE
19
20          :SIZE FOR CLOCK
21 005146  004737 022500   JSR    PC,SIZCLK        ;SEE IF CLOCK PRESENT
22 005152  000413           BR    5$                ;YES - CLOCK IS PRESENT
23 005154           4$:
24 005154  104000           EMT
25 005156  104401 005164   TYPE  ,65$              ;;TYPE ASCIZ STRING
26 005162  000405           BR    64$              ;;GET OVER THE ASCIZ
27 005176           ;;65$: .ASCIZ <CRLF>/PROG HLT/
28 005176  000000           64$: HALT                  ;PROGRAM HALT !!
29 005200  000765           BR    4$
30 005202           5$:
31 005202  005737 000042   .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
32 005206  001012           TST   @#42              ;;ARE WE RUNNING UNDER XXDP/ACT?
33 005210  123727 001242 000001  BNE   66$              ;;BRANCH IF YES
34 005216  001406           CMPB  $ENV,#1           ;;ARE WE RUNNING UNDER APT?
35 005220  023727 001154 000176  BEQ   66$              ;;BRANCH IF YES
36 005226  001005           CMP   SWR,#SWREG       ;;SOFTWARE SWITCH REG SELECTED?
37 005230  104407           BNE   67$              ;;BRANCH IF NO
38 005232  000403           GTSWR                  ;;GET SOFT-SWR SETTINGS
39 005234  112737 000001 001150 66$:  MOVB  #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
40 005242  005242           67$:
41 005242  000240           READY: NOP              ;READY TO START TEST
42 005244  105737 001300           TSTB  $DEVN            ;ANY DRIVES IN MAP ?
43 005250  001007           BNE   2$                ;BR IF YES
44 005252  005737 000042           TST   @#42            ;ANY MONITOR PRESENT ?
45 005256  001002           BNE   1$                ;BR IF YES
46 005260  000137 002716           JMP   START            ;JUMP TO START
47 005264  000137 021356           1$:   JMP   $EOP            ;RETURN CONTROL TO MONITOR
48
49 005270  105037 001116           2$:   CLRB  $TSTNM          ;RESET TEST NUMBER
50 005274  005037 001206           CLR   $TIMES           ;INITIALIZE NUMBER OF ITERATIONS
51 005300  004737 027012           JSR   PC,$TKINT        ;INITIALIZE TTY
52 005304  012746 000300           MOV   #PR6,-(SP)       ;;PUT NEW PS ON STACK
53 005310  012746 005316           MOV   #64$,-(SP)      ;;PUT NEW PC ON STACK
54 005314  000002           RTI                    ;;POP NEW PC AND PS

```

```
005316
41 005316 117737 174140 001234 64$:      MOVB    @TSTQUE,$UNIT    ;LOAD UNIT NUMBER
42
43                                     ;CLEAR MASSBUS CONTROLLER, SELECT DRIVE AND DETERMINE THE LAST TRACK
44                                     ;OF THE DIFFERENT DRIVE TYPES
45 005324 012737 002000 001330      MOV     #TA4,LSTRK      ;ASSUME LAST TRACK FOR RM02/3 = 4.
46 005332 013700 001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
47 005336 012760 000040 000010      MOV     #CLR, RMCS2(R0) ;CLEAR MASSBUS
48 005344 117760 174112 000010      MOVB   @TSTQUE, RMCS2(R0) ;SELECT DEVICE UNDER TEST
49 005352 016002 000026              MOV     RMDT(R0),R2     ;GET RMDT AND
50 005356 042702 177770              BIC    #177770,R2      ;SAVE DRIVE TYPE BITS
51 005362 022702 000007              CMP     #7,R2          ;IS IT AN RM05 ?
52 005366 001003                    BNE    3$              ;NO, MUST BE AN RM02 OR RM03
53 005370 012737 011000 001330      MOV     #TA16!TA2,LSTRK ;YES--SET LAST TRACK = 18.
54 005376
55 3$:
```

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

.SBTTL REGISTER AND STORAGE USAGE

:REGISTER ASSIGNMENTS

:R0 = UNIBUS ADDRESS OF RH CONTROLLER
:R1 = ADDRESS OF ENTRY IN TEST QUE CORRESPONDING TO THE
UNIT UNDER TEST
:R2,R3 = WORKING REGISTERS FOR TEST IN PROGRESS, MUST BE
SAVED BY SUBROUTINES
:R4,R5 = GENERAL WORKING REGISTERS, ARE NOT SAVED BY
SUBROUTINES
:R6 = STACK POINTER
:R7 = LINKAGE REGISTER TO SUBROUTINES

:STORAGE ASSIGNMENTS

:\$TMP0-\$TMP4 TEMPORARY STORAGE, NOT SAVED BY SUBROUTINES
:\$GDDAT,\$BDDAT EXPECTED AND RECEIVED STATUS FOR ERROR TYPEOUT
:\$GDADR,\$BDADR ADDRESS OF EXPECTED AND RECEIVED STATUS IF APPLICABLE,
ALSO THE ADDRESS OF A REGISTER ERROR
:
:\$STN = TEST NUMBER
:\$UNIT = NUMBER OF DEVICE BEING TESTED
:\$GINBF = THE REGISTER INPUT BUFFER HAS A STORAGE LOCATION FOR
EACH REGISTER, AND IS USED WHEN READING STATUS AND
CONTROL DATA
:\$GOTBF = THE REGISTER OUTPUT BUFFER HAS A STORAGE LOCATION FOR
EACH REGISTER, AND IS USED FOR ASSEMBLING DATA TO BE
WRITTEN IN REGISTERS
:
:

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

*TEST 1 TRANSFER TEST

TST1:

005376 000004 SCOPE ;SCOPE CALL
005376 000240 NOP
005400 012700 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
005406 013700 001276 MOV \$BASE,R0 ;R0 = UNIBUS ADDRESS
005412 013701 001462 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
005416 012737 000001 001226 MOV #1,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

MOV #0,R2 ;R2 = REGISTER INDEX

;CLEAR THE MASSBUS AND VERIFY THAT NONEXISTANT DEVICE ERROR IS RESET
10\$:

005430 004737 023334 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
005430 016037 000010 001142 MOV RMCS2(R0),\$BDDAT ;STORE RMCS2 AT \$BDDAT
005442 032737 010000 001142 BIT #NED,\$BDDAT
005450 001417 BEQ 20\$
005452 111137 001140 MOV (R1),\$GDDAT
005456 042737 177770 001140 BIC #^CUNTMSK,\$GDDAT
005464 052737 000100 001140 BIS #IR,\$GDDAT
005472 010037 001136 MOV R0,\$BDADR
005476 062737 000010 001136 ADD #RMCS2,\$BDADR
005504 104001 EMT 1
005506 000475 BR 60\$

;READ THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE READ
DOES NOT SET 'NED' ERROR
20\$:

005510 010003 MOV R0,R3 ;R3 = REGISTER ADDRESS
005510 060203 ADD R2,R3
005512 011304 MOV (R3),R4 ;READ REGISTER
005516 032760 010000 000010 BIT #NED,RMCS2(R0) ;IS 'NED' SET??
005524 001470 BEQ 70\$;NO!!

;GO CLEAR CONTROLLER
STORE RMCS2 AT \$BDDAT

005526 004737 023334 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
005532 016037 000010 001142 MOV RMCS2(R0),\$BDDAT ;STORE RMCS2 AT \$BDDAT
005540 032737 010000 001142 BIT #NED,\$BDDAT
005546 001417 BEQ 30\$
005550 111137 001140 MOV (R1),\$GDDAT
005554 042737 177770 001140 BIC #^CUNTMSK,\$GDDAT
005562 052737 000100 001140 BIS #IR,\$GDDAT
005570 010037 001136 MOV R0,\$BDADR
005574 062737 000010 001136 ADD #RMCS2,\$BDADR
005602 104001 EMT 1
005604 000436 BR 60\$

;WRITE THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE WRITE
DOES NOT SET 'NED' ERROR
30\$:

005606 012713 000000 MOV #0,(R3) ;WRITE REGISTER
005612 032760 010000 000010 BIT #NED,RMCS2(R0) ;IS 'NED' SET??
005620 001432 BEQ 70\$;NO!!

;COULD NOT READ OR WRITE THE REGISTER WITHOUT SETING 'NED' ERROR -

```

48 ;ADVANCE THE REGISTER INDEX AND REPEAT THE TEST FOR THE NEXT
49 ;AVAILABLE DEVICE REGISTER
50 005622 40$:
51 005622 062702 000002 ADD #2,R2 ;ADVANCE TO NEXT REGISTER
52 005626 022702 000002 CMP #RMWC,R2 ;IS THIS RMWC??
53 005632 001773 BEQ 40$ ;YES - TRY NEXT REGISTER
54 005634 022702 000004 CMP #RMBA,R2 ;IS THIS RMBA??
55 005640 001770 BEQ 40$ ;YES - TRY NEXT REGISTER
56 005642 022702 000010 CMP #RMCS2,R2 ;IS THIS RMCS2??
57 005646 001765 BEQ 40$ ;YES - TRY ANOTHER REGISTER
58 005650 022702 000016 CMP #RMAS,R2 ;IS THIS RMAS ??
59 005654 001762 BEQ 40$ ;YES - TRY ANOTHER REGISTER
60 005656 022702 000022 CMP #RMDB,R2 ;IS THIS RMDB??
61 005662 001757 BEQ 40$ ;YES - TRY ANOTHER REGISTER
62 005664 022702 000046 CMP #RMEC2,R2 ;IS THIS A LEGAL REGISTER
63 005670 103257 BHIS 10$ ;YES - TRY THIS REGISTER
64
65 ;GOT 'NONEXISTENT DEVICE' ERROR FOR EVERY REMOTE REGISTER ADDRESS
66 005672 50$:
67 005672 013737 001276 001136 MOV $BASE,$BDADR ;STORE BASE ADDRESS
68 005700 104002 EMT 2
69 005702 000137 021320 60$: JMP $EOSP ;GO SELECT NEXT DEVICE
70
71 005706 70$:
72
73 ;*****
; *TEST 2 CTOD TEST
;*****
;TST2:
005706 SCOPE ;SCOPE CALL
005706 000004 NOP
005710 000240
005712 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
005716 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
005722 013701 001462 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
005726 012737 000002 001226 MOV #2,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
74
75 005734 004737 023334 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
76
77 ;WRITE ONES IN REMOTE REGISTERS
78 005740 012760 000076 000000 MOV #ILF76,RMCS1(R0) ;LOAD RMCS1
79 005746 012760 177777 000006 MOV #-1,RMDA(R0) ;LOAD RMDA
80 005754 012760 001777 000034 MOV #CYLMSK,RMDC(R0) ;LOAD RMDC
81 005762 012760 016200 000032 MOV #^CXNUOF,RMOF(R0) ;LOAD RMOF
82
83 ;READ REMOTE REGISTERS TWICE
84 005770 012702 000001 MOV #1,R2
85 005774 10$:
005774 016037 000000 001332 MOV RMCS1(R0),RMCS1I ;STORE RMCS1 IN INPUT BUFFER
86 006002 016037 000006 001340 MOV RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
87 006010 016037 000034 001366 MOV RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
88 006016 016037 000032 001364 MOV RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
89 006024 005302 DEC R2
90 006026 100362 BPL . 10$
91
92 ;SEE IF ANY ONE BITS CAME BACK
93 006030 042737 177701 001332 BIC #^CILF76,RMCS1I ;IS RMCS1 0??
  
```

```
94 006036 001014 BNE 20$ ;NO!!
95 006040 005737 001340 TST RMDAI ;IS RMDA 0??
96 006044 001011 BNE 20$ ;NO!!
97 006046 042737 176000 001366 BIC #XNUDC,RMDCI ;IS RMDC 0??
98 006054 001005 BNE 20$ ;NO!!
99 006056 042737 161577 001364 BIC #XNUOF,RMOFI ;IS RMOF 0 ??
100 006064 001001 BNE 20$ ;NO!!
101
102 ;CANNOT READ/WRITE A ONF FROM ANY REMOTE REGISTER
103 006066 104003 EMT 3
104 006070 20$:
105
106 ;*****
;*TEST 3 MASSBUS INITIALIZE TEST
;*****
TST3:
006070 SCOPE ;SCOPE CALL
006070 000004 NOP
006072 000240 MOV #STACK,SP ;LOAD THE STACK POINTER
006074 012706 001100 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
006100 013700 001276 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
006104 013701 001462 MOV #3,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
006110 012737 000003 001226 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
107
108 006116 004737 023334 ;WRITE ONES IN SELECTED REGISTERS
109
110 MOV #11,F76,RMCS1(R0) ;LOAD RMCS1
111 006122 012760 000076 000000 MOV #-1,RMER1(R0) ;LOAD RMER1
112 006130 012760 177777 000014 MOV #-1,RMER2(R0) ;LOAD RMER2
113 006136 012760 177777 000042
114
115 ;INITIALIZE MASSBUS WITH A CLEAR
116 006144 004737 023334 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
117
118 ;READ THE REGISTERS THAT WERE WRITTEN
119 006150 016037 000000 001332 MOV RMCS1(R0),RMCS1I ;STORE RMCS1 IN INPUT BUFFER
120 006156 016037 000014 001346 MOV RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
121 006164 016037 000042 001374 MOV RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
122
123 ;SEE IF ANY REGISTER BITS WERE CLEARED
124 006172 052737 177701 001332 BIS #^CILF76,RMCS1I ;SET ANY BIT NOT WRITTEN
125 006200 052737 001567 001374 BIS #XNUER2,RMER2I
126 006206 022737 177777 001332 CMP #-1,RMCS1I ;ANY ZEROS IN RMCS1??
127 006214 001011 BNE 10$ ;YES!!
128 006216 022737 177777 001346 CMP #-1,RMER1I ;ANY ZEROS IN RMER1??
129 006224 001005 BNE 10$ ;YES!!
130 006226 022737 177777 001374 CMP #-1,RMER2I ;ANY ZEROS IN RMER2??
131 006234 001001 BNE 10$
132
133 ;NONE OF THE BITS WERE CLEARED
134 006236 104004 EMT 4
135 006240 10$:
136
137 ;*****
;*TEST 4 CLEAR STUCK ACTIVE TEST
;*****
```

```

006240          TST4:
006240 000004          SCOPE          ;SCOPE CALL
006242 000240          NOP
006244 012706 001100  MOV      #STACK,SP      ;LOAD THE STACK POINTER
006250 013700 001276  MOV      $BASE,R0        ;R0 = UNIBUS ADDRESS
006254 013701 001462  MOV      TSTQUE,R1       ;R1 = POINTER TO DEVICE
006260 012737 000004 001226  MOV      #4,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
138
139 006266 004737 023334  JSR      PC,CNTCLR      ;GO CLEAR CONTROLLER
140
141          ;WRITE ONES IN TEST REGISTERS
142 006272 012760 177777 000014  MOV      #-1,RMER1(R0)  ;LOAD RMER1
143 006300 012760 177777 000042  MOV      #-1,RMER2(R0)  ;LOAD RMER2
144 006306 012760 000001 000024  MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
145
146          ;READ TEST REGISTERS AND SEE IF ANY BITS ARE ON
147 006314 016037 000014 001346  MOV      RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
148 006322 016037 000042 001374  MOV      RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
149 006330 016037 000024 001356  MOV      RMMR1(R0),RMMR1I ;STORE RMMR1 IN INPUT BUFFER
150 006336 042737 040000 001346  BIC      #UNS,RMER1I     ;DONT ACCEPT UNSAFE
151 006344 001011          BNE      10$            ;BRANCH IF ANY OTHER BITS ON
152 006346 042737 040200 001374  BIC      #SKI!DVC,RMER2I ;DONT ACCEPT SKI OR DVC
153 006354 001005          BNE      10$            ;BRANCH IF ANY OTHER BITS ON
154 006356 032737 000001 001356  BIT      #DMD,RMMR1I     ;BRANCH IF DMD IS ON
155 006364 001001          BNE      10$
156 006366 104026          EMT      26
157 006370          10$:
158
159          ;*****
          ;*TEST 5          TRISTATE TRANSFER TEST
          ;*****

```

```

006370          TST5:
006370 000004          SCOPE          ;SCOPE CALL
006372 000240          NOP
006374 012706 001100  MOV      #STACK,SP      ;LOAD THE STACK POINTER
006400 013700 001276  MOV      $BASE,R0        ;R0 = UNIBUS ADDRESS
006404 013701 001462  MOV      TSTQUE,R1       ;R1 = POINTER TO DEVICE
006410 012737 000005 001226  MOV      #5,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
160
161 006416 005002          CLR      R2              ;CLEAR ERROR FLAGS
162 006420 004737 023334  JSR      PC,CNTCLR      ;GO CLEAR CONTROLLER
163
164          ;WRITE ONES IN SELECTED REGISTERS
165 006424 012760 000076 000000  MOV      #ILF76,RMCS1(R0) ;LOAD RMCS1
166 006432 012760 177777 000006  MOV      #-1,RMDA(R0)    ;LOAD RMDA
167 006440 012760 177777 000014  MOV      #-1,RMER1(R0)  ;LOAD RMER1
168 006446 012760 177777 000032  MOV      #-1,RMOF(R0)   ;LOAD RMOF
169 006454 012760 177777 000042  MOV      #-1,RMER2(R0)  ;LOAD RMER2
170
171          ;WRITE ZEROS IN SELECTED REGISTERS
172 006462 012760 000000 000000  MOV      #0,RMCS1(R0)   ;LOAD RMCS1
173 006470 012760 000000 000006  MOV      #0,RMDA(R0)   ;LOAD RMDA
174 006476 012760 000000 000014  MOV      #0,RMER1(R0)  ;LOAD RMER1
175 006504 012760 000000 000032  MOV      #0,RMOF(R0)   ;LOAD RMOF
176 006512 012760 000000 000034  MOV      #0,RMDC(R0)   ;LOAD RMDC
177 006520 012760 000000 000042  MOV      #0,RMER2(R0)  ;LOAD RMER2

```

```

178
179 ;READ BACK ALL REGISTERS
180 006526 016037 000000 001332 MOV RMCS1(R0),RMCS1I ;STORE RMCS1 IN INPUT BUFFER
181 006534 016037 000006 001340 MOV RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
182 006542 016037 000014 001346 MOV RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
183 006550 016037 000032 001364 MOV RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
184 006556 016037 000034 001366 MOV RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
185 006564 016037 000042 001374 MOV RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
186
187 ;CHECK EACH REGISTER CONTENT FOR ZERO BITS WRITTEN & READ
188 006572 012702 177777 MOV #-1,R2 ;ACCUMULATE ZEROS IN R2
189 006576 052737 177701 001332 BIS #^C1LF76,RMCS1I ;SET ALL BITS NOT WRITTEN
190 006604 052737 161577 001364 BIS #XNUOF,RMOFI
191 006612 052737 176000 001366 BIS #XNUDC,RMDCI
192 006620 052737 001567 001374 BIS #XNUER2,RMER2I
193 006626 005137 001332 COM RMCS1I ;COMPLEMENT REGISTER CONTENTS
194 006632 005137 001340 COM RMDAI
195 006636 005137 001346 COM RMER1I
196 006642 005137 001364 COM RMOFI
197 006646 005137 001366 COM RMDCI
198 006652 005137 001374 COM RMER2I
199 006656 043702 001332 BIC RMCS1I,R2 ;ACCUMULATE ALL ZERO BITS
200 006662 043702 001340 BIC RMDAI,R2
201 006666 043702 001346 BIC RMER1I,R2
202 006672 043702 001364 BIC RMOFI,R2
203 006676 043702 001366 BIC RMDCI,R2
204 006702 043702 001374 BIC RMER2I,R2
205 006706 001407 BEQ 10$ ;BRANCH IF EACH BIT IS ZERO
206
207 ;ONE OR MORE BIT POSITIONS ARE NOT ZERO
208 006710 010237 001142 MOV R2,$BDDAT ;SAVE RESULT FOR TYPE
209 006714 005037 001140 CLR $GDDAT ;LOAD EXPECTED RESULT
210 006720 104005 EMT 5
211 006722 052702 000001 BIS #BIT0,R2 ;SET ERROR FLAG
212
213 006726 10$: JSR PC,CNTCLR ;GO CLEAR CONTROLLER
006726 004737 023334
214
215 ;PRESET SELECTED REGISTERS TO ZEROS
216 ;(ASSUME RMCS1, RMER1, RMER2 WERE CLEARED BY INIT)
217 006732 012760 000000 000006 MOV #0,RMDA(R0) ;LOAD RMDA
218 006740 012760 000000 000032 MOV #0,RMOF(R0) ;LOAD RMOF
219 006746 012760 000000 000034 MOV #0,RMDC(R0) ;LOAD RMDC
220
221 ;WRITE ONES IN SELECTED REGISTERS
222 006754 012760 000076 000000 MOV #1LF76,RMCS1(R0) ;LOAD RMCS1
223 006762 012760 177777 000006 MOV #-1,RMDA(R0) ;LOAD RMDA
224 006770 012760 016200 000032 MOV #^CXNUOF,RMOF(R0) ;LOAD RMOF
225 006776 012760 001777 000034 MOV #^CXNUDC,RMDC(R0) ;LOAD RMDC
226 007004 012760 177777 000014 MOV #-1,RMER1(R0) ;LOAD RMER1
227 007012 012760 176210 000042 MOV #^CXNUER2,RMER2(R0) ;LOAD RMER2
228
229 ;READ ALL REGISTERS
230 007020 016037 000000 001332 MOV RMCS1(R0),RMCS1I ;STORE RMCS1 IN INPUT BUFFER
231 007026 016037 000006 001340 MOV RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
232 007034 016037 000032 001364 MOV RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
233 007042 016037 000034 001366 MOV RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER

```



```

234 007050 016037 000014 001346      MOV      RMER1(R0),RMER1I      ;STORE RMER1 IN INPUT BUFFER
235 007056 016037 000042 001374      MOV      RMER2(R0),RMER2I      ;STORE RMER2 IN INPUT BUFFER
236
237                                     ;CHECK EACH REGISTER CONTENT FOR ONE BITS WRITTEN & READ
238 007064 042737 177701 001332      BIC      #^CILF76,RMCS1I      ;CLEAR ALL BITS NOT WRITTEN
239 007072 042737 161577 001364      BIC      #XNUOF,RMOFI
240 007100 042737 176000 001366      BIC      #XNUDC,RMDCI
241 007106 042737 001567 001374      BIC      #XNUER2,RMER2I
242 007114 005002                                     CLR      R2                    ;ACCUMULATE ONES IN R2
243 007116 053702 001332      BIS      RMCS1I,R2            ;ACCUMULATE ALL ONE BITS
244 007122 053702 001340      BIS      RMDAI,R2
245 007126 053702 001364      BIS      RMOFI,R2
246 007132 053702 001366      BIS      RMDCI,R2
247 007136 053702 001346      BIS      RMER1I,R2
248 007142 053702 001374      BIS      RMER2I,R2
249 007146 022702 177777      CMP      #-1,R2              ;SEE IF EACH BIT POSITION WAS ONE
250 007152 001410      BEQ      20$                  ;BRANCH IF NONE STUCK
251
252                                     ;ONE OR MORE BIT POSITIONS ARE NOT ONE
253 007154 010237 001142      MOV      R2,$BDDAT           ;SAVE RESULT FOR TYPE
254 007160 012737 177777 001140      MOV      #-1,$GDDAT         ;EXPECTED RESULT
255 007166 104006      EMT      6
256 007170 052702 000002      BIS      #BIT1,R2           ;SET ERROR FLAG
257 007174      20$:
258 007174 005702      TST      R2                  ;ANY ERRORS DETECTED ??
259 007176 001126      BNE      30$                 ;YES - DONT DO BIT TEST
260 007200 012702 000001      MOV      #1,R2              ;R2=BIT POSITION
261
262 007204      25$:
263 007204 004737 023334      JSR      PC,CNTCLR          ;GO CLEAR CONTROLLER
264
265                                     ;WRITE THE BIT PATTERN IN SELECTED DEVICE REGISTERS
266 007210 010260 000006      MOV      R2,RMDA(R0)        ;LOAD RMDA
267 007214 010260 000032      MOV      R2,RMOF(R0)        ;LOAD RMOF
268 007220 010260 000034      MOV      R2,RMDC(R0)        ;LOAD RMDC
269 007224 010260 000014      MOV      R2,RMER1(R0)       ;LOAD RMER1
270 007230 010260 000042      MOV      R2,RMER2(R0)       ;LOAD RMER2
271
272                                     ;READ BACK THE REGISTERS
273 007234 016037 000006 001340      MOV      RMDA(R0),RMDAI     ;STORE RMDA IN INPUT BUFFER
274 007242 016037 000032 001364      MOV      RMOF(R0),RMOFI     ;STORE RMOF IN INPUT BUFFER
275 007250 016037 000034 001366      MOV      RMDC(R0),RMDCI     ;STORE RMDC IN INPUT BUFFER
276 007256 016037 000014 001346      MOV      RMER1(R0),RMER1I   ;STORE RMER1 IN INPUT BUFFER
277 007264 016037 000042 001374      MOV      RMER2(R0),RMER2I   ;STORE RMER2 IN INPUT BUFFER
278
279                                     ;CHECK REGISTER CONTENTS FOR CORRECT PATTERN
280 007272 005003      CLR      R3                  ;R3=ACCUMULATED ONE BIT
281 007274 012704 177777      MOV      #-1,R4             ;R4=ACCUMULATED ZERO BITS
282 007300 013705 001340      MOV      RMDAI,R5           ;GET ANY GOOD BITS FROM RMDA
283 007304 050503      BIS      R5,R3
284 007306 005105      COM      R5
285 007310 040504      BIC      R5,R4
286 007312 013705 001364      MOV      RMOFI,R5           ;GET GOOD BITS FROM RMOF
287 007316 042705 161577      BIC      #XNUOF,R5
288 007322 050503      BIS      R5,R3
289 007324 005105      COM      R5
290 007326 042705 161577      BIC      #XNUOF,R5
    
```

```

290 007332 040504      BIC    R5,R4
291 007334 013705 001366  MOV    RMDCI,R5      ;GET GOOD BITS FROM RMDC
292 007340 042705 176000  BIC    #XNUDC,R5
293 007344 050503      BIS    R5,R3
294 007346 005105      COM    R5
295 007350 042705 176000  BIC    #XNUDC,R5
296 007354 040504      BIC    R5,R4
297 007356 013705 001346  MOV    RMER1,R5      ;GET GOOD BITS FROM RMER1
298 007362 050503      BIS    R5,R3
299 007364 005105      COM    R5
300 007366 040504      BIC    R5,R4
301 007370 013705 001374  MOV    RMER2,R5      ;GET GOOD BITS FROM RMER2
302 007374 042705 001567  BIC    #XNUER2,R5
303 007400 050503      BIS    R5,R3
304 007402 005105      COM    R5
305 007404 042705 001567  BIC    #XNUER2,R5
306 007410 040504      BIC    R5,R4
307 007412 010205      MOV    R2,R5      ;RESET ALL ONES IN R3 EXCEPT
308 007414 005105      COM    R5      ;FOR THE TEST BIT
309 007416 040503      BIC    R5,R3
310 007420 040204      BIC    R2,R4      ;RESET TEST BIT IN R4
311 007422 050403      BIS    R4,R3      ;COMBINE ACCUMULATED 1'S + 0'S
312 007424 020302      CMP    R3,R2      ;IS PATTERN OK??
313 007426 001406      BEQ    26$      ;YES!!
314 007430 010237 001140  MOV    R2,$GDDAT      ;SAVE TEST PATTERN
315 007434 010337 001142  MOV    R3,$BDDAT      ;SAVE RESULT
316 007440 104007      EMT    7
317 007442 000404      BR     30$      ;SKIP TO NEXT

```

```

318
319 ;ADVANCE R2 TO THE NEXT PATTERN AND REPEAT TEST
320 26$:

```

```

321 007444 006302      ASL    R2      ;SHIFT THE BIT
322 007446 001402      BEQ    30$      ;EXIT IF DONE
323 007450 000137 007204  JMP    25$
324 007454

```

```

325 30$:
326

```

```

;*****
;*TEST 6 REGISTER SELECT TEST

```

```

;*****
TST6:

```

```

007454      SCOPE      ;SCOPE CALL
007454 000004      NOP
007456 000240      MOV    #STACK,SP      ;LOAD THE STACK POINTER
007460 012706 001100  MOV    $BASE,R0      ;R0 = UNIBUS ADDRESS
007464 013700 001276  MOV    TSTQUE,R1      ;R1 = POINTER TO DEVICE
007470 013701 001462  MOV    #6,$TESTN      ;:SET TEST NUMBER IN APT MAIL BOX
007474 012737 000006 001226  MOV

```

```

327
328 ;THE FOLLOWING TABLE GIVES MASSBUS REGISTER SELECT VALUES FOR
329 ;EACH DEVICE REGISTER
330

```

REGISTER NAME	REG SEL (16,8,4,2,1)
RMCS1	00000
RMDS	00001
RMER1	00010

```

337 : RMMR1 00011
338 : RMAS 00100
339 : RMDA 00101
340 : RMDT 00110
341 : RMLA 00111
342 : RMSN 01000
343 : RMOF 01001
344 : RMDC 01010
345 : RMHR 01011
346 : RMMR2 01100
347 : RMER2 01101
348 : RMEC1 01110
349 : RMEC2 01111
350 :

```

```

: EACH REGISTER SELECT LINE IS TESTED FOR A STUCK AT ONE,
: STUCK AT ZERO FAULT. AS AN EXAMPLE, TO TEST REG SEL 1.
: FOR S-A-0, RMER1 IS WRITTEN WITH ZEROS. THEN THE REGISTER
: THAT HAS THE SAME SELECT VALUE, EXCEPT FOR THE SELECT LINE
: BEING TESTED, IS WRITTEN WITH ONES. IN THIS EXAMPLE,
: RMMR1 IS WRITTEN WITH ONES. IF SELECT LINE 1 IS S-A-0,
: THE ALL ONES WORD WILL BE WRITTEN IN RMER1, AND RMER1
: WILL NOT BE 0 WHEN READ BACK.

```

```

360 007502 005002
361 007504 012703 177777
362
363
364

```

```

CLR R2 ;R2= ZEROS SOURCE
MOV #-1,R3 ;R3= ONES SOURCE

```

:TEST REG SEL 1 FOR S-A-0

```

365 007510 004737 023334
366 007514 010260 000014
367 007520 010260 000034
368 007524 010360 000024
369 007530 010360 000036
370 007534 016037 000014 001346
371 007542 016037 000034 001366
372 007550 020337 001346
373 007554 001007
374 007556 052737 176000 001366
375 007564 020337 001366
376 007570 001001
377 007572 104710
378
379

```

```

JSR PC,CNTCLR ;GO CLEAR CONTROLLER
MOV R2,RMER1(R0) ;LOAD RMER1
MOV R2,RMDC(R0) ;LOAD RMDC
MOV R3,RMMR1(R0) ;LOAD RMMR1
MOV R3,RMHR(R0) ;LOAD RMHR
MOV RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
MOV RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
CMP R3,RMER1I
BNE 10$
BIS #XNUDC,RMDCI
CMP R3,RMDCI
BNE 10$
EMT 10

```

:TEST REG SEL 1 FOR S-A-1
10\$:

```

380 007574
381 007574 004737 023334
382 007600 010260 000006
383 007604 010260 000032
384 007610 010260 000042
385 007614 010360 000016
386 007620 010360 000030
387 007624 010360 000040
388 007630 016037 000006 001340
389 007636 016037 000032 001364
390 007644 016037 000042 001374
391 007652 020337 001340
392 007656 001015
393 007660 052737 161577 001364

```

```

JSR PC,CNTCLR ;GO CLEAR CONTROLLER
MOV R2,RMDA(R0) ;LOAD RMDA
MOV R2,RMOF(R0) ;LOAD RMOF
MOV R2,RMER2(R0) ;LOAD RMER2
MOV R3,RMAS(R0) ;LOAD RMAS
MOV R3,RMSN(R0) ;LOAD RMSN
MOV R3,RMMR2(R0) ;LOAD RMMR2
MOV RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
MOV RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
MOV RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
CMP R3,RMDAI
BNE 20$
BIS #XNUOF,RMOFI

```

394 007666 020337 001364
395 007672 001007
396 007674 052737 001567
397 007702 020337 001374
398 007706 001001
399 007710 104011

CMP R3,RMOF I
BNE 20\$
BIS #XNUER2,RMER2I
CMP R3,RMER2I
BNE 20\$
EMT 11

001374

400
401
402 007712

:TEST REG SEL 2 FOR S-A-0
20\$:

403 007712 004737 023334
404 007716 010260 000006
405 007722 010260 000032
406 007726 010260 000042
407 007732 010360 000020
408 007736 010360 000036
409 007742 010360 000046
410 007746 016037 000006
411 007754 016037 000032
412 007762 016037 000042
413 007770 020337 001340
414 007774 001015
415 007776 052737 161577
416 010004 020337 001364
417 010010 001007
418 010012 052737 001567
419 010020 020337 001374
420 010024 001001
421 010026 104012

JSR PC,CNTCLR ;GO CLEAR CONTROLLER
MOV R2,RMDA(R0) ;LOAD RMDA
MOV R2,RMOF(R0) ;LOAD RMOF
MOV R2,RMER2(R0) ;LOAD RMER2
MOV R3,RMLA(R0) ;LOAD RMLA
MOV R3,RMHR(R0) ;LOAD RMHR
MOV R3,RMEC2(R0) ;LOAD RMEC2
MOV RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
MOV RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
MOV RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
CMP R3,RMDAI
BNE 30\$
BIS #XNUOF,RMOF I
CMP R3,RMOF I
BNE 30\$
BIS #XNUER2,RMER2I
CMP R3,RMER2I
BNE 30\$
EMT 12

001340

001364

001374

001364

001374

422
423
424 010030

:TEST REG SEL 2 FOR S-A-1
30\$:

425 010030 004737 023334
426 010034 010260 000014
427 010040 010260 000034
428 010044 012760 000076
429 010052 010360 000030
430 010056 016037 000014
431 010064 016037 000034
432 010072 052737 177701
433 010100 020337 001346
434 010104 001007
435 010106 052737 176000
436 010114 020337 001366
437 010120 001001
438 010122 104013

JSR PC,CNTCLR ;GO CLEAR CONTROLLER
MOV R2,RMER1(R0) ;LOAD RMER1
MOV R2,RMDC(R0) ;LOAD RMDC
MOV #ILF76,RMCS1(R0) ;LOAD RMCS1
MOV R3,RMSN(R0) ;LOAD RMSN
MOV RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
MOV RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
BIS #^CILF76,RMER1I
CMP R3,RMER1I
BNE 40\$
BIS #XNUDC,RMDC I
CMP R3,RMDC I
BNE 40\$
EMT 13

000000

001346

001366

001346

001366

439
440
441 010124

:TEST REG SEL 4 FOR S-A-0
40\$:

442 010124 004737 023334
443 010130 010260 000014
444 010134 010260 000032
445 010140 010260 000034
446 010144 010360 000026
447 010150 010360 000042
448 010154 010360 000044
449 010160 016037 000014
450 010166 016037 000032

JSR PC,CNTCLR ;GO CLEAR CONTROLLER
MOV R2,RMER1(R0) ;LOAD RMER1
MOV R2,RMOF(R0) ;LOAD RMOF
MOV R2,RMDC(R0) ;LOAD RMDC
MOV R3,RMDT(R0) ;LOAD RMDT
MOV R3,RMER2(R0) ;LOAD RMER2
MOV R3,RMEC1(R0) ;LOAD RMEC1
MOV RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
MOV RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER

001346

001364

```

451 010174 016037 000034 001366      MOV      RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
452 010202 020337 001346      CMP      R3,RMER1I
453 010206 001015                    BNE     50$
454 010210 052737 161577 001364      BIS     #XNUOF,RMOFI
455 010216 020337 001364      CMP      R3,RMOFI
456 010222 001007                    BNE     50$
457 010224 052737 176000 001366      BIS     #XNUDC,RMDCI
458 010232 020337 001366      CMP      R3,RMDCI
459 010236 001001                    BNE     50$
460 010240 104014                    EMT     14
461
462
463 010242                    ;TEST REG SEL 4 FOR S-A-1
464 010242 004737 023334 50$:      JSR     PC,CNTCLR ;GO CLEAR CONTROLLER
465 010246 010260 000006      MOV     R2,RMDA(R0) ;LOAD RMDA
466 010252 010260 000042      MOV     R2,RMER2(R0) ;LOAD RMER2
467 010256 010360 000012      MOV     R3,RMDS(R0) ;LOAD RMDS
468 010262 010360 000032      MOV     R3,RMOF(R0) ;LOAD RMOF
469 010266 016037 000006 001340      MOV     RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
470 010274 016037 000042 001374      MOV     RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
471 010302 020337 001340      CMP     R3,RMDAI
472 010306 001007                    BNE     60$
473 010310 052737 001567 001374      BIS     #XNUER2,RMER2I
474 010316 020337 001374      CMP     R3,RMER2I
475 010322 001001                    BNE     60$
476 010324 104015                    EMT     15
477
478
479 010326                    ;TEST REG SEL 8 FOR S-A-0
480 010326 004737 023334 60$:      JSR     PC,CNTCLR ;GO CLEAR CONTROLLER
481 010332 010260 000014      MOV     R2,RMER1(R0) ;LOAD RMER1
482 010336 010260 000006      MOV     R2,RMDA(R0) ;LOAD RMDA
483 010342 010360 000034      MOV     R3,RMDC(R0) ;LOAD RMDC
484 010346 010360 000042      MOV     R3,RMER2(R0) ;LOAD RMER2
485 010352 016037 000014 001346      MOV     RMER1(R0),RMER1I ;STORE RMER1 IN INPUT BUFFER
486 010360 016037 000006 001340      MOV     RMDA(R0),RMDAI ;STORE RMDA IN INPUT BUFFER
487 010366 020337 001346      CMP     R3,RMER1I
488 010372 001004                    BNE     70$
489 010374 020337 001340      CMP     R3,RMDAI
490 010400 001001                    BNE     70$
491 010402 104016                    EMT     16
492
493
494 010404                    ;TEST REG SEL 8 FOR S-A-1
495 010404 004737 023334 70$:      JSR     PC,CNTCLR ;GO CLEAR CONTROLLER
496 010410 010260 000032      MOV     R2,RMOF(R0) ;LOAD RMOF
497 010414 010260 000034      MOV     R2,RMDC(R0) ;LOAD RMDC
498 010420 010260 000042      MOV     R2,RMER2(R0) ;LOAD RMER2
499 010424 010360 000012      MOV     R3,RMDS(R0) ;LOAD RMDS
500 010430 010360 000014      MOV     R3,RMER1(R0) ;LOAD RMER1
501 010434 010360 000006      MOV     R3,RMDA(R0) ;LOAD RMDA
502 010440 016037 000032 001364      MOV     RMOF(R0),RMOFI ;STORE RMOF IN INPUT BUFFER
503 010446 016037 000034 001366      MOV     RMDC(R0),RMDCI ;STORE RMDC IN INPUT BUFFER
504 010454 016037 000042 001374      MOV     RMER2(R0),RMER2I ;STORE RMER2 IN INPUT BUFFER
505 010462 052737 161577 001364      BIS     #XNUOF,RMOFI
506 010470 001015                    BNE     80$
507 010472 022737 176000 001366      CMP     #XNUDC,RMDCI

```

508 010500 020337 001366
509 010504 001007
510 010506 052737 001567 001374
511 010514 020337 001374
512 010520 001001
513 010522 104017
514 010524
515
516
517
518

CMP R3,RMDCI
BNE 80\$
BIS #XNUER2,RMER2I
CMP R3,RMER2I
BNE 80\$
EMT 17

80\$:

;REGISTER SELECT 16 IS TESTED BY THE ILR TEST

::*****
;*TEST 7 DRIVE TYPE TEST

::*****
TST7:

010524
010524 000004
010526 000240
010530 012706 001100
010534 013700 001276
010540 013701 001462
010544 012737 000007 001226
519
520 010552 016002 000026
521 010556 022702 020024
522 010562 001437
523 010564 022702 024024
524 010570 001434
525
526 010572 022702 020025
527 010576 001431
528 010600 022702 024025
529 010604 001426
530
531 010606 022702 020027
532 010612 001423
533 010614 022702 024027
534 010620 001420
535
536 010622 010237 001142
537 010626 012737 020024 001174
538 010634 012737 024024 001176
539 010642 010037 001136
540 010646 062737 000026 001136
541 010654 104057
542 010656 000137 021320
543 010662
544
545

SCOPE ;SCOPE CALL
NOP
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV \$BASE,R0 ;R0 = UNIBUS ADDRESS
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
MOV #7,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV RMDT(R0),R2 ;STORE RMDT AT R2
CMP #SNGPRT,R2 ;SINGLE PORT RM03 ?
BEQ 10\$;YES !!
CMP #DULPRT,R2 ;DUAL PORT RM03 ?
BEQ 10\$;YES !!
CMP #SNGPRT!BIT0,R2 ;SINGLE PORT RM02 ?
BEQ 10\$;YES !!
CMP #DULPRT!BIT0,R2 ;DUAL PORT RM02 ?
BEQ 10\$;YES !!
CMP #SNGPRT!BIT1!BIT0,R2 ;SINGLE PORT RM05 ?
BEQ 10\$;YES !!
CMP #DULPRT!BIT1!BIT0,R2 ;DUAL PORT RM05 ?
BEQ 10\$
MOV R2,\$BDDAT ;GET RECIEVED DRIVE TYPE
MOV #SNGPRT,\$TMP0 ;GET EXPECTED DRIVE TYPE
MOV #DULPRT,\$TMP1
MOV R0,\$BDADR ;LOAD BAD ADDRESS
ADD #RMDT,\$BDADR
EMT 57
JMP \$EOSP ;GO TO NEXT DEVICE

10\$:

::*****
;*TEST 10 DEVICE AVAILABLE TEST

::*****
TST10:

010662
010662 000004
010664 000240
010666 012706 001100
010672 013700 001276
010676 013701 001462

SCOPE ;SCOPE CALL
NOP
MOV #STACK,SP ;LOAD THE STACK POINTER
MOV \$BASE,R0 ;R0 = UNIBUS ADDRESS
MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE

```

546 010702 012737 000010 001226      MOV      #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
547 010710 004737 023334              JSR      PC,CNTCLR      ;GO CLEAR CONTROLLER
548 010714 016037 000000 001142      MOV      RMCS1(R0),$BDDAT ;STORE RMCS1 AT $BDDAT
549 010722 042737 173777 001142      BIC      #^CDVA,$BDDAT  ;CLEAR ALL BUT DVA
550 010730 001006              BNE      10$           ;BRANCH IF DVA SET
551 010732 012737 004000 001140      MOV      #DVA,$GDDAT   ;SETUP EXPECTED
552 010740 010037 001136              MOV      R0,$BDADR    ;SETUP REG ADDRESS
553 010744 104060              EMT      60
554 010746              10$:
555
556
;*****
;*TEST 11      SEARCH TIMEOUT TEST
;*****
TST11:
010746
010746 000004              SCOPE                    ;SCOPE CALL
010750 000240              NOP
010752 012706 001100      MOV      #STACK,SP     ;LOAD THE STACK POINTER
010756 013700 001276      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
010762 013701 001462      MOV      TSTQUE,R1     ;R1 = POINTER TO DEVICE
010766 012737 000011 001226      MOV      #11,$TESTN   ;;SET TEST NUMBER IN APT MAIL BOX
557
558
559 010774 012737 000000 001414      ;LOAD REGISTER OUTPUT BUFFER WITH COMMAND PARAMETERS
560 011002 012737 000000 001442      MOV      #0,RMDAO      ;SECTOR=0=TRACK
561 011010 012737 010000 001440      MOV      #0,RMDCO      ;CYLINDER=0
562 011016 012737 054244 001412      MOV      #FMT16,RMOFO  ;16 BIT FORMAT
563 011024 012737 177777 001410      MOV      #BUFFER,RMBAO ;STARTING BUFFER ADDRESS
564 011032 012737 000061 001406      MOV      #-1,RMWCO     ;WORD COUNT
565
566 011040 004737 023430              JSR      PC,ENBSCH     ;EXECUTE DATA COMMAND TO POINT WHERE
;SEARCH IS ENABLED USING SUBROUTINE.
011044 000402              BR      10$           ;BRANCH TO 10$ IF NO ERROR
011046 104000
567 011050 000462              BR      50$           ;SLOC REST PF TEST
568
569
570 011052              ;START THE CLOCK AND WAIT FOR 100 MS
571 011052 012737 000144 001530      10$:      MOV      #100,WATCH    ;SET WATCHDOG TIMER VALUE
011060 004777 170446              JSR      PC,@CLOCK     ;START THE CLOCK
572 011064 005737 001530      20$:      TST      WATCH
573 011070 001375              BNE      20$
574 011072 004777 170436              JSR      PC,@STOP     ;STOP THE CLOCK
575
576
577 011076 016037 000014 001142      ;VERIFY THAT OPI IS NOT SET (SEARCH TIMEOUT IS DISABLED)
578 011104 010037 001136              MOV      RMER1(R0),$BDDAT ;STORE RMER1 AT $BDDAT
579 011110 062737 000014 001136              MOV      R0,$BDADR    ;SET UP FOR ERROR MSG
580 011116 042737 157777 001142              ADD      #RMER1,$BDADR
581 011124 001404              BIC      #^COPI,$BDDAT
582 011126 005037 001140              BEQ      30$           ;BRANCH IF NO ERROR
583 011132 104020              CLR      $GDDAT
584
585 011134 000430              EMT      20
586
587
BR      50$           ;DISABLED
;SKIP
;ENABLE SEARCH TIMEOUT, THEN WAIT 100 MS

```

```

T11 SEARCH TIMEOUT TEST
588 011136
589 011136 012760 041401 000024 30$: MOV #DMD!MUR!DBEN!MOC,RMMR1(R0) ;LOAD RMMR1
590 011144 012737 000144 001530 MOV #100.,WATCH ;SET WATCHDOG TIMER VALUE
011152 004777 170354 JSR PC,@CLOCK ;START THE CLOCK
591 011156 005737 001530 40$: TST WATCH
592 011162 001375 BNE 40$
593 011164 004777 170344 JSR PC,@STOP ;STOP THE CLOCK
594
595 ;OPI SHOULD NOW BE SET (SEARCH TIMEOUT IS ENABLED)
596 011170 016037 000014 001142 MOV RMER1(R0),SBDDAT ;STORE RMER1 AT SBDDAT
597 011176 042737 157777 001142 BIC #^COPI,SBDDAT
598 011204 001004 BNE 50$
599 011206 012737 020000 001140 MOV #OPI,$GDDAT
600 011214 104021 EMT 21
601 011216 50$:
602
603 ;:*****
;*TEST 12 SET DTE TEST
;:*****
TST12:
011216 000004 SCOPE ;SCOPE CALL
011220 000240 NOP
011222 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
011226 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
011232 013701 001462 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
011236 012737 000012 001226 MOV #12,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
604
605 011244 010037 001136 MOV R0,$BDADR ;SETUP ERROR MSG
606 011250 062737 000014 001136 ADD #RMER1,$BDADR
607 011256 005037 001140 CLR $GDDAT
608
609 ;INITILAIIZE AND VERIFY THAT DRIVE TIMING ERROR IS RESET
610 011262 004737 023334 JSR PC,CNTCLR ;GO CLEAR CONTROLLER
611
612 011266 016037 000014 001142 MOV RMER1(R0),SBDDAT ;STORE RMER1 AT SBDDAT
613 011274 042737 167777 001142 BIC #^CDTE,SBDDAT
614 011302 001402 BEQ 10$ ;BRANCH IF DTE=0
615 011304 104031 EMT 31
616 011306 000517 BR 50$ ;SKIP REST OF TEST
617
618 ;SET MAINTENANCE INDEX PULSE AND VERIFY DTE REMAINS RESET
619 011310 10$:
620 011310 012760 000001 000024 MOV #DMD,RMMR1(R0) ;LOAD RMMR1
621 011316 012760 000005 000024 MOV #DMD!MI,RMMR1(R0) ;LOAD RMMR1
622 011324 016037 000014 001142 MOV RMER1(R0),SBDDAT ;STORE RMER1 AT SBDDAT
623 011332 042737 167777 001142 BIC #^CDTE,SBDDAT
624 011340 001402 BEQ 20$
625 011342 104000 EMT
626 011344 000500 BR 50$ ;COMPARE SHOULD BE RESET
627
628 ;EXECUTE DUMMY DATA COMMAND TO ENABLE SEARCH
629 011346 20$:
630 011346 012737 000000 001414 MOV #0,RMDAO
631 011354 012737 000005 001442 MOV #5,RMDCO
632 011362 012737 010000 001440 MOV #FMT16,RMOFO
633 011370 012737 054244 001412 MOV #BUFFER,RMBAO

```



```

634 011376 012737 177777 001410      MOV      #-1,RMWCO
635 011404 012737 000061 001406      MOV      #WD!GO,RMCS10
636
637 011412 004737 023430      JSR      PC,ENBSCH      ;EXECUTE DATA COMMAND TO POINT WHERE
                          ;SEARCH IS ENABLED USING SUBROUTINE.
                          ;BRANCH TO 30$ IF NO ERROR
      011416 000402      BR       30$
      011420 104000      EMT
638 011422 000451      BR       50$
639

```

```

640      ;WITH SEARCH ENABLED, SET AND RESET SECTOR PULSE TO SET ENABLE
641      ;SEARCH FLOP.
642 011424      30$:

```

```

643 011424 012760 051441 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MS,RMMR1(R0) ;LOAD RMMR1
644 011432 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
645

```

```

646      ;SET SECTOR PULSE AND VERIFY DTE DOES NOT SET
647      ;
648      ;
649      ;

```

```

650 011440 016037 000014 001142      MOV      RMER1(R0), $BDDAT      ;STORE RMER1 AT $BDDAT
651 011446 042737 167777 001142      BIC      #^CDTE,$BDDAT
652 011454 001402      BEQ     40$
653 011456 104023      EMT     23
654 011460 001432      BEQ     50$      ;COMPARE SHOULD BE RESET
655

```

```

656      ;FORCE SECTOR COMPARE
657 011462      40$:

```

```

658 011462 012760 051403 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
659 011470 012760 051443 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MSC!MS,RMMR1(R0) ;LOAD RMMR1
660 011476 012760 051403 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
661

```

```

662      ;SET SECTOR PULSE AND VERIFY DTE SETS
663 011504 012760 051441 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MS,RMMR1(R0) ;LOAD RMMR1
664 011512 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
665 011520 016037 000014 001142      MOV      RMER1(R0), $BDDAT      ;STORE RMER1 AT $BDDAT
666 011526 042737 167777 001142      BIC      #^CDTE,$BDDAT
667 011534 001004      BNE     50$
668 011536 012737 010000 001142      MOV      #DTE,$BDDAT
669 011544 104024      EMT     24
670

```

```

671 011546      50$:      ;SECTOR COMPARE SET
672
673

```

```

:*****
:*TEST 13      FORMAT CHANGE TEST

```

```

:*****
:TST13:

```

```

011546      SCOPE      ;SCOPE CALL
011546 000004      NOP
011550 000240      MOV      #STACK,SP      ;LOAD THE STACK POINTER
011552 012706 001100      MOV      $BASE,R0      ;R0 = UNIBUS ADDRESS
011556 013700 001276      MOV      TSTQUE,R1      ;R1 = POINTER TO DEVICE
011562 013701 001462      MOV      #13,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
011566 012737 000013 001226      MOV      #50$,R2      ;R2=TABLE POINTER
674
675 011574 012702 012124
676
677

```

```

;INITILAIZE AND SET THE FORMAT BIT, USE INDEX PULSE TO CLEAR FORMAT CHANGE

```

```

678 011600
679 011600 004737 023334
680 011604 011260 000032
681 011610 012760 000001 000024
682 011616 012760 000005 000024
683
684
685 011624 012737 000000 001414
686 011632 012737 000005 001442
687 011640 016237 000002 001440
688 011646 012737 054244 001412
689 011654 012737 177777 001410
690 011662 012737 000061 001406
691
692 011670 004737 023430
    011674 000402
    011676 104000
693 011700 000514
694
695
696
697 011702
698 011702 012760 051403 000024
699 011710 012760 051443 000024
700 011716 012760 051403 000024
701 011724 012760 051401 000024
702
703
704 011732 016037 000014 001142
705 011740 042737 167777 001142
706 011746 001416
707 011750 010037 001136
708 011754 062737 000014 001136
709 011762 005037 001140
710 011766 011237 001174
711 011772 016237 000002 001176
712 012000 104025
713 012002 000453
714
715
716 012004
717 012004 012760 051405 000024
718
719
720 012012 012760 051403 000024
721 012020 012760 051443 000024
722 012026 012760 051403 000024
723
724
725 012034 012760 051441 000024
726 012042 012760 051401 000024
727 012050 016037 000014 001142
728 012056 042737 167777 001142
729 012064 001012
730 012066 010037 001136
731 012072 062737 000014 001136

10$:
    JSR    PC,CNTCLR      ;GO CLEAR CONTROLLER
    MOV    (R2),RMOF(R0)  ;LOAD FORMAT MODE IN RMOF
    IOV    #DMD,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MI,RMMR1(R0) ;LOAD RMMR1

;SETUP AND EXECUTE DUMMY DATA COMMAND USING OPPOSITE FORMAT
    MOV    #0,RMDAO
    MOV    #5,RMDCO
    MOV    2(R2),RMOFO
    MOV    #BUFFER,RMBAO
    MOV    #-1,RMWCO
    MOV    #WD!GO,RMCS10

    JSR    PC,ENBSCH      ;EXECUTE DATA COMMAND TO POINT WHERE
                        ;SEARCH IS ENABLED USING SUBROUTINE.
                        ;BRANCH TO 20$ IF NO ERROR
    BR     20$
    EMT
    BR     60$

;FORMAT CHANGE FLOP SHOULD BE SET - VERIFY BY TRYING TO FORCE A
;DRIVE TIMING ERROR WHICH SHOULD NOT SET.
20$:
    MOV    #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MUR!DBEN!MOC!DTO!MSC!MS,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1

;VERIFY THAT DRIVE TIMING ERROR DIDNT SET
    MOV    RMER1(R0),SBDDAT ;STORE RMER1 AT SBDDAT
    BIC    #^CDTE,SBDDAT
    BEQ    30$
    MOV    R0,$BDADR ;SETUP ERROR MESSAGE
    ADD    #RMER1,$BDADR
    CLR    $GDDAT
    MOV    (R2),$TMP0
    MOV    2(R2),$TMP1
    EMT    25
    BR     60$ ;A FORMAT CHANGE

;CLEAR THE FORMAT CHANGE FLOP W/INDEX PULSE
30$:
    MOV    #DMD!MUR!DBEN!MOC!DTO!MI,RMMR1(R0) ;LOAD RMMR1

;ENABLE SEARCH AND FORCE SECTOR COMPARE
    MOV    #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MUR!DBEN!MOC!DTO!MSC!MS,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1

;SET DTE W/ANOTHER SECTOR PULSE - VERIFY DTE IS SET
    MOV    #DMD!MUR!DBEN!MOC!DTO!MS,RMMR1(R0) ;LOAD RMMR1
    MOV    #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
    MOV    RMER1(R0),SBDDAT ;STORE RMER1 AT SBDDAT
    BIC    #^CDTE,SBDDAT
    BNE    40$
    MOV    R0,$BDADR ;SETUP ERROR MESSAGE
    ADD    #RMER1,$BDADR

```

```

732 012100 012737 010000 001140      MOV      #DTE,$GDDAT
733 012106 104027                      EMT      27
734 012110 000410                      BR       60$
735
736                                     :DO TEST W/18 TO 16 FORMAT CHANGE DURING SECOND EXECUTION
737 012112                                     40$:
738 012112 022702 C12124                CMP      #50$,R2
739 012116 001005                      BNE     60$                ;BRANCH IF DONE TEST
740 012120 005722                      TST     (R2)+             ;MOVE POINTER
741 012122 000626                      BR      10$              ;REPEAT TEST
742
743 012124 010000                50$:      .WORD   FMT16           ;16-18 FORMAT CHANGE
744 012126 000000                .WORD   0                ;18-16 FORMAT CHANGE
745 012130 010000                .WORD   FMT16
746 012132                60$:                                     ;END OF TEST
747
748                                     :*****
:TEST 14      PROM STROBE TEST
:*****
TST14:
012132                SCOPE                ;SCOPE CALL
012132 000004                NOP
012134 000240                MOV      #STACK,SP       ;LOAD THE STACK POINTER
012136 012706 001100                MOV      $BASE,R0        ;R0 = UNIBUS ADDRESS
012142 013700 001276                MOV      TSTQUE,R1      ;R1 = POINTER TO DEVICE
012146 013701 001462                MOV      #14,$TESTN     ;:SET TEST NUMBER IN APT MAIL BOX
012152 012737 000014 001226
749
750 012160 004737 023334                JSR     PC,CNTCLR        ;GO CLEAR CONTROLLER
751 012164 010037 001136                MOV     RO,$BDADR       ;SETUP ERROR MESSAGE
752 012170 062737 000024 001136                ADD     #RMMR1,$BDADR
753
754                                     :SET DIAGNOSTIC MODE AND TRY TO RESET PROM STROBE
755 012176 012760 000001 000024                MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
756 012204 012702 000021                MOV     #17.,R2        ;R2=MAXIMUM # CLOCK PULSES
757 012210                5$:      MOV     #DMD!MCLK,RMMR1(R0) ;LOAD RMMR1
012210 012760 004001 000024                MOV     #DMD,RMMR1(R0) ;LOAD RMMR1
758 012216 012760 000001 000024                MOV     RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT
759 012224 016037 000024 001142                BIC     #^CWC,$BDDAT
760 012232 042737 177737 001142                BEQ     6$
761 012240 001406                DEC     R2
762 012242 005302                BNE     5$
763 012244 001361                CLR     $GDDAT
764 012246 005037 001140                EMT     60$
765 012252 104000                BR      60$
766 012254 000501                6$:      MOV     #17.,R2
767 012256 012702 000021                10$:
768 012262                MOV     #DMD!MCLK,RMMR1(R0) ;LOAD RMMR1
012262 012760 004001 000024                MOV     #DMD,RMMR1(R0) ;LOAD RMMR1
769 012270 012760 000001 000024                MOV     RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT
770 012276 016037 000024 001142                BIC     #^CWC,$BDDAT
771 012304 042737 177737 001142                BEQ     20$
772 012312 001007                BNE     20$                ;EXIT L OP WHEN PROM STROBE ON
773 012314 005302                DEC     R2
774 012316 001361                BNE     10$
775 012320 012737 000040 001140                MOV     #WC,$GDDAT
776 012326 104030                EMT     30

```

```

777 012330 000453 BR 60$
778
779 ;VERIFY PROM STROBE IS ON FOR 3 BIT CLOCKS
780 012332 012702 000003 20$:
781 012332 012702 000003 MOV #3.,R2
782 012336 012760 004001 000024 25$:
783 012344 012760 000001 000024 MOV #DMD!MCLK,RMMR1(R0) ;LOAD RMMR1
784 012352 016037 000024 001142 MOV #DMD,RMMR1(R0) ;LOAD RMMR1
785 012360 042737 177737 001142 MOV RMMR1(R0),SBDDAT ;STORE RMMR1 AT SBDDAT
786 012366 001005 BIC #^CWC,SBDDAT
787 BNE 30$ ;BRANCH IF PROM STORBE DID NOT
;DROP EARLY
788 012370 012737 000040 001140 MOV #WC,$GDDAT
789 012376 104032 EMT 32
790 012400 000427 BR 60$
791 012402 005302 30$: DEC R2
792 012404 001354 BNE 25$
793
794 ;VERIFY PROM STROBE IS OFF FOR 8 BIT CLOCKS
795 012406 012702 000010 40$:
796 012412 012760 004001 000024 MOV #DMD!MCLK,RMMR1(R0) ;LOAD RMMR1
797 012420 012760 000001 000024 MOV #DMD,RMMR1(R0) ;LOAD RMMR1
798 012426 016037 000024 001142 MOV RMMR1(R0),SBDDAT ;STORE RMMR1 AT SBDDAT
799 012434 042737 177737 001142 BIC #^CWC,SBDDAT
800 012442 001404 BEQ 50$ ;BRANCH IF PROM STROBE
801 012444 005037 001140 CLR $GDDAT ;DID NOT SET EARLY
802 012450 104033 EMT 33
803 012452 000402 BR 60$
804 012454 005302 50$: DEC R2
805 012456 001355 BNE 40$
806 012460 60$:
807
808 ;*****
;*TEST 15 SYNC WORD COUNT INHIBIT TEST
;*****
;TST15:
012460 000004 SCOPE ;SCOPE CALL
012462 000240 NOP
012464 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
012470 013700 001276 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
012474 013701 001462 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
012500 012737 000015 001226 MOV #15,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
809
810 ;SETUP REGISTER OUTPUT BUFFER FOR SUBROUTINES
811 012506 012737 000005 001442 MOV #5,RMDCO ;CYLINDER=5
812 012514 012737 000000 001414 MOV #0,RMDAO ;SECTOR=0, TRACK=0
813 012522 012737 010000 001440 MOV #FMT16,RMOFO ;16 BIT FORMAT
814 012530 012737 054244 001412 MOV #BUFFER,RMBAO ;STARTING BUFFER ADDRESS
815 012536 012737 177777 001410 MOV #-1,RMWCO ;WORD COUNT
816 012544 012737 000061 001406 MOV #WD!GO,RMCS10 ;WRITE DATA COMMAND
817
818 012552 004737 023430 JSR PC,ENBSCH ;EXECUTE DATA COMMAND TO POINT WHERE
;SEARCH IS ENABLED USING SUBROUTINE.
012556 000402 BR 10$ ;BRANCH TO 10$ IF NO ERROR
012560 104000 EMT
    
```

```
819 012562 000562 BR 120$ ;SKIP IF FAILURE
820 0-2564 10$:
821 012564 004737 024302 JSR PC,SCTCMP ;FORCE SECTOR COMPARE USING SUBROUTINE
012570 000402 BR 20$ ;BRANCH TO 20$ IF NO ERROR
012572 104000 EMT
822 012574 000555 BR 120$
823
824 ;CAN NOW STEP DATA TIMING SEQUENCER WITH SECTOR COMPARE SET
825
826 ;STEP THE SEQUENCER TO LOCATION 10(10) WHILE CHECKING 'PLFS'
827 012576 012702 000000 20$: MOV #0,R2 ;R2=SEQUENCER ADDRESS
828
829 ;PULSE MAINTENANCE CLOCK UNTIL PROM STROBE SETS
830 012602 30$:
831 012602 012760 055401 000024 MOV #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0) ;LOAD RMMR1
832 012610 012760 051401 000024 MOV #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
833
834 ;SEE IF PROM STROBE IS SET
835 012616 016003 000024 MOV RMMR1(R0),R3 ;STORE RMMR1 AT R3
836 012622 032703 000040 BIT #WC,R3
837 012626 001765 BEQ 30$ ;ISSUE NEXT BIT CLOCK
838 012630 005202 INC R2 ;INCREMENT SEQUENCER ADDRESS
839
840 ;CHECK 'LOOKING FOR SYNC' - SHOULD BE ZERO UNTIL LOCATION 11
841 012632 010337 001142 MOV R3,$BDDAT
842 012636 042737 175777 001142 BIC #^CPLFS,$BDDAT
843 012644 001413 BEQ 50$ ;BRANCH IF PLFS IS ZERO
844 012646 005037 001140 CLR $GDDAT
845 012652 010037 001136 MOV R0,$BDADR
846 012656 062737 000024 001136 ADD #RMMR1,$BDADR
847 012664 010237 001174 MOV R2,$TMP0
848 012670 104034 EMT 34
849 012672 000516 BR 120$ ;SKIP REST OF TEST
850
851 ;EXIT LOOP IF SEQUENCER NOW AT LOCATION 10
852 012674 50$:
853 012674 022702 000012 CMP #10.,R2
854 012700 001414 BEQ 70$
855
856 ;PULSE MAINTENANCE CLOCK UNTIL PROM STROBE RESETS
857 012702 60$:
858 012702 012760 055401 000024 MOV #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0) ;LOAD RMMR1
859 012710 012760 051401 000024 MOV #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
860
861 ;SEE IF PROM STROBE IS RESET
862 012716 016003 000024 MOV RMMR1(R0),R3 ;STORE RMMR1 AT R3
863 012722 032703 000040 BIT #WC,R3
864 012726 001365 BNE 60$
865 012730 000724 BR 30$ ;GO STEP SEQUENCER TO NEXT LOC
866
867 ;THE NEXT PROM STROBE SHOULD SET 'PLFS' AT LOCATION 11(10) OF THE
868 ;DATA TIMING SEQUENCER.
869 012732 70$:
870 012732 012702 000020 MOV #16.,R2 ;R2=NUMBER OF BIT CLOCKS
871
872 ;ISSUE 16 BIT CLOCKS TO GET NEXT PROM STROBE
873 012736 80$:
```

```

874 012736 012760 055401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0) ;LOAD RMMR1
875 012744 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
876 012752 005302      DEC      R2
877 012754 001370      BNE      80$
878
879      ;VERIFY THAT 'PLFS' IS NOW SET
880 012756 016037 000024 001142      MOV      RMMR1(R0),SBDDAT ;STORE RMMR1 AT SBDDAT
881 012764 042737 175777 001142      BIC      #^CPLFS,SBDDAT
882 012772 001012      BNE      90$ ;BRANCH IF PLFS IS SET
883 012774 012737 002000 001140      MOV      #PLFS,$GDDAT ;SETUP ERROR MESSAGE
884 013002 010037 001136      MOV      R0,$BDADR
885 013006 062737 000024 001136      ADD      #RMMR1,$BDADR
886 013014 104035      EMT      35
887 013016 000444      BR       120$ ;SKIP REST OF TEST
888
889      ;ISSUE 3 MORE BIT CLOCKS TO RESET PROM STROBE
890 013020      90$:
891 013020 012702 000003      MOV      #3,R2
892 013024      95$:
893 013024 012760 055401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0) ;LOAD RMMR1
894 013032 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
895 013040 005302      DEC      R2
896 013042 001370      BNE      95$
897
898      ;WITH 'LOOKING FOR SYNC SET, FURTHER BIT CLOCKS SHOULD NOT SET
899 013044 012702 000040      ;PROM STROBE
900      MOV      #32.,R2 ;R2=NUMBER OF BIT CLOCKS
901
902      ;PULSE BIT CLOCK AND VERIFY PROM STROBE DOES NOT SET
903 013050      100$:
904 013050 012760 055401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0) ;LOAD RMMR1
905 013056 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
906 013064 016003 000024      MOV      RMMR1(R0),R3 ;STORE RMMR1 AT R3
907 013070 042703 177737      BIC      #^CWC,R3
908 013074 001413      BEQ      110$ ;BRANCH IF PROM STROBE IS 0
909 013076 005037 001140      CLR      $GDDAT ;SETUP ERROR MESSAGE
910 013102 010337 001142      MOV      R3,$BDDAT
911 013106 010037 001136      MOV      R0,$BDADR
912 013112 062737 000024 001136      ADD      #RMMR1,$BDADR
913 013120 104036      EMT      36
914 013122 000402      BR       120$
915 013124 005302      110$: DEC      R2 ;CONTINUE FOR 32 BIT CLOCKS
916 013126 001350      BNE      100$
917 013130      120$: ;END OF TEST
918
919      ;*****
920      ;*TEST 16 SYNC DETECTION TEST
921      ;*****
922      ;*****
923      ;TST16:
924      SCOPE ;SCOPE CALL
925      NOP
926      MOV      #STACK,SP ;LOAD THE STACK POINTER
927      MOV      $BASE,R0 ;R0 = UNIBUS ADDRESS
928      MOV      TSTQUE,R1 ;R1 = POINTER TO DEVICE
929      MOV      #16,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
  
```

```

013130
013130 000004
013132 000240
013134 012706 001100
013140 013700 001276
013144 013701 001462
013150 012737 000016 001226
  
```

```
920  
921 ;SETUP REGISTER OUTPUT BUFFER FOR SUBROUTINES  
922 013156 012737 000000 001414 MOV #0,RMDAO ;SECTOR 0,TRACK 0  
923 013164 012737 000005 001442 MOV #5,RMDCO ;CYLINDER 5  
924 013172 012737 010000 001440 MOV #FMT16,RMOFO ;16 BIT MODE  
925 013200 012737 054244 001412 MOV #BUFFER,RMBAO ;BUFFER ADDRESS  
926 013206 012737 177777 001410 MOV #-1,RMWCO ;CORD COUNT  
927 013214 012737 000071 001406 MOV #RD!GO,RMCS10 ;READ DATA COMMAND  
928  
929 013222 004737 023430 JSR PC,ENBSCH ;EXECUTE DATA COMMAND TO POINT WHERE  
 ;SEARCH IS ENABLED USING SUBROUTINE.  
 ;BRANCH TO 10$ IF NO ERROR  
 ; 3226 000402 BR 10$  
 ; 3230 104000 EMT  
930 013232 000550 BR 100$  
931 013234 10$:  
932 013234 004737 024302 JSR PC,SCTCMP ;FORCE SECTOR COMPARE USING SUBROUTINE  
 ;BRANCH TO 20$ IF NO ERROR  
 ; 3240 000402 BR 20$  
 ; 3242 104000 EMT  
933 013244 000543 BR 100$  
934 013246 20$:  
935 013246 004737 024410 JSR PC,SETLFS ;SET 'LOOKING FOR SYNC' USING SUBROUTINE  
 ;BRANCH TO 30$ IF NO ERROR  
 ; 3252 000402 BR 30$  
 ; 3254 104000 EMT  
936 013256 000536 BR 100$  
937  
938 ;CLOCK ALL ONES SYNC PATTERN AND VERIFY SYNC IS NOT DETECTED  
939 ;(USING PROM STORBE AS INDICATION)  
940 013260 30$:  
941 013260 012702 000020 MOV #16.,R2 ;NUMBER OF ONE BITS  
942 013264 010037 001136 MOV R0,$BDADR ;SETUP ERROR MESSAGE  
943 013270 062737 000024 001136 ADD #RMMR1,$BDADR  
944 013276 005037 001140 CLR $GDDAT  
945 013302 012760 053401 000024 MOV #MR1AAA!MRD,RMMR1(R0) ;LOAD RMMR1  
946 013310 40$:  
947 013310 012760 057401 000024 MOV #MR1AAA!MRD!MCLK,RMMR1(R0) ;LOAD RMMR1  
948 013316 012760 053401 000024 MOV #MR1AAA!MRD,RMMR1(R0) ;LOAD RMMR1  
949 013324 016037 000024 001142 MOV RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT  
950 013332 042737 177737 001142 BIC #^CWC,$BDDAT  
951 013340 001405 BEQ 50$  
952 013342 012737 177777 001174 MOV #177777,$TMP0  
953 013350 104037 EMT  
954 013352 000500 BR 100$  
955 013354 50$:  
956 013354 005302 DEC R2 ;REPEAT FOR 16 BITS  
957 013356 001354 BNE 40$  
958 013360 012702 000020 MOV #16.,R2 ;NUMBER OF ZERO BITS  
959  
960 ;CLOCK ALL ZERO SYNC PATTERN AND VERIFY SYNC IS NOT DETECTED  
961 013364 60$:  
962 013364 012760 055401 000024 MOV #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1  
963 013372 012760 051401 000024 MOV #MR1AAA,RMMR1(R0) ;LOAD RMMR1  
964 013400 016037 000024 001142 MOV RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT  
965 013406 042737 177737 001142 BIC #^CWC,$BDDAT ;MAKE SURE SYNC NOT DETECTED  
966 013414 001404 BEQ 70$  
967 013416 005037 001174 CLR $TMP0  
968 013422 104037 EMT  
969 013424 000453 BR 100$ ;SKIP IF FAILURE
```

```

970 013426
971 013426 005302
972 013430 001355
973 013432 012702 000040
974 013436 012703 000031
975
976
977
978 013442 012737 051401 001432
979 013450 000241
980 013452 006003
981 013454 103003
982 013456 052737 002000 001432
983 013464
    013464 013760 001432 000024
984 013472 052737 004000 001432
985 013500 013760 001432 000024
986 013506 042737 004000 001432
987 013514 013760 001432 000024
988 013522 016037 000024 001142
989 013530 042737 177737 001142
990 013536 001006
991 013540 005302
992 013542 001337
993 013544 012737 000040 001140
994 013552 104040
995 013554
996
997
    70$:
        DEC R2 ;REPEAT FOR 16 BITS
        BNE 60$
        MOV #32,R2 ;R2=NUMBER OF BITS
        MOV #000031,R3 ;R3=SYNC PATTERN FOR LEFT SHIFT

    :CLOCK THE BINARY SYNC PATTERN (10011000) AND VERIFY THAT SYNC IS
    :DETECTED
    80$: MOV #MR1AAA,RMMR10 ;GENERATE VALUE OF RMMR1
        CLC ;CLEAR THE CARR
        ROR R3 ;SHIFT RIGHT
        BCC 90$ ;BRANCH IF C BIT CLEAR
        BIS #MRD,RMMR10 ;SET MRD IF PATTERN BIT SETS
    90$: MOV RMMR10,RMMR1(R0) ;LOAD RMMR1
        BIS #MCLK,RMMR10 ;SET THE MAINTENANCE DATA CLK
        MOV RMMR10,RMMR1(R0) ;LOAD RMMR1
        BIC #MCLK,RMMR10 ;RESET BIT CLOCK
        MOV RMMR10,RMMR1(R0) ;LOAD RMMR1
        MOV RMMR1(R0),$BDDAT ;STORE RMMR1 AT $BDDAT
        BIC #*CWC,$BDDAT
        BNE 100$ ;BRANCH IF PROM STROBE IS SET
        DEC R2 ;CONTINUE FOR 16 BIT CLOCKS
        BNE 80$
        MOV #WC,$GDDAT
        EMT 40
    100$: ;END OF TEST
    
```

 ;*TEST 17 ABORT SYNC DETECTION TEST

```

    013554
    013554 000004
    013556 000240
    013560 012706 001100
    013564 013700 001276
    013570 013701 001462
    013574 012737 000017 001226
998
999
1000 013602 012737 000000 001414
1001 013610 012737 000005 001442
1002 013616 012737 010000 001440
1003 013624 012737 054244 001412
1004 013632 012737 177777 001410
1005 013640 012737 000071 001406
1006
1007 013646 004737 023430
    JSR PC,ENBSCH ;EXECUTE DATA COMMAND TO POINT WHERE
    ;SEARCH IS ENABLED USING SUBROUTINE.
    BR 10$ ;BRANCH TO 10$ IF NO ERROR
    EMT
    BR 50$
    10$: JSR PC,SCTCMP ;FORCE SECTOR COMPARE USING SUBROUTINE
        BR 20$ ;BRANCH TO 20$ IF NO ERROR
        EMT
    TST17:
        SCOPE ;SCOPE CALL
        NOP
        MOV #STACK,SP ;LOAD THE STACK POINTER
        MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
        MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
        MOV #17,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
    ;SETUP REGISTER OUTPUT BUFFER FOR SUBROUTINES
        MOV #0,RMDAO ;SECTOR 0, TRACK 0
        MOV #5,RMDCO ;CYLINDER 5
        MOV #FMT16,RMOFO ;16 BIT FORMAT
        MOV #BUFFER,RMBAO ;STARTING BUFFER ADDRESS
        MOV #-1,RMWCO ;WORD COUNT
        MOV #RD!GO,RMCS10 ;READ DATA COMMAND
    
```



```
1011 013670 000442 BR 50$
1012 013672 20$:
1013 013672 004737 024410 JSR PC,SETLFS ;SET 'LOOKING FOR SYNC' USING SUBROUTINE
    013676 000402 BR 30$ ;BRANCH TO 30$ IF NO ERROR
    013700 104000 EMT
1014 013702 000435 BR 50$
1015
1016 ;LOOKING FOR SYNC INHIBITS WORD CLOCK, HOWEVER, 'DRIVE TIMING ERROR'
1017 ;SHOULD RESET 'LFS WORD COUNT INHIBIT' FLOP
1018 ;FORCE DRIVE TIMING ERROR
1019 013704 30$:
1020 013704 012760 051441 000024 MOV #MR1AAA!MS,RMMR1(R0) ;LOAD RMMR1
1021
1022 ;PULSE BIT CLOCK AND VERIFY PROM STROBE SETS
1023 013712 012702 000020 MOV #16.,R2 ;R2, NUMBER OF BIT CLOCKS
1024 013716 40$:
1025 013716 012760 055441 000024 MOV #MR1AAA!MS!MCLK,RMMR1(R0) ;LOAD RMMR1
1026 013724 012760 051441 000024 MOV #MR1AAA!MS,RMMR1(R0) ;LOAD RMMR1
1027 013732 016037 000024 001142 MOV RMMR1(R0), $BDDAT ;STORE RMMR1 AT $BDDAT
1028 013740 042737 177737 001142 BIC #^CWC,$BDDAT
1029 013746 001013 BNE 50$
1030 013750 005302 DEC R2
1031 013752 001361 BNE 40$
1032 013754 012737 000040 001140 MOV #WC,$GDDAT
1033 013762 010037 001136 MOV R0,$BDADR
1034 013766 062737 000024 001136 ADD #RMMR1,$BDADR
1035 013774 104041 EMT 41
1036 013776 50$:
1037
1038
1039
```

```
*****
;*TEST 20 SYNC GENERATION TEST
*****
```

```
013776
013776 000004 TST20:
014000 000240 SCOPE ;SCOPE CALL
014002 012706 001100 NOP
014006 013700 001276 MOV #STACK,SP ;LOAD THE STACK POINTER
014012 013701 001462 MOV $BASE,R0 ;R0 = UNIBUS ADDRESS
014016 012737 000020 001226 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
    MOV #20,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
1040
1041 ;SETUP REGISTER OUTPUT BUFFER FOR SUBROUTINES
1042 014024 012737 000000 001414 MOV #0,RMDAO ;SECTOR 0, TRACK 0
1043 014032 012737 000005 001442 MOV #5,RMDCO ;CYL 5
1044 014040 012737 010000 001440 MOV #FMT16,RMOFO ;16 BIT MODE
1045 014046 012737 054244 001412 MOV #BUFFER,RMBAO ;BUFFER ADDRESS
1046 014054 012737 177776 001410 MOV #-2,RMWCO ;WORD COUNT
1047 014062 012737 000063 001406 MOV #WH!GO,RMCS10 ;WHITE HEAD AND DATA COM
1048
1049 014070 004737 023430 JSR PC,ENBSCH ;EXECUTE DATA COMMAND TO POINT WHERE
    014074 000403 BR 10$ ;SEARCH IS ENABLED USING SUBROUTINE.
    014076 104000 EMT ;BRANCH TO 10$ IF NO ERROR
1050 014100 000137 014470 JMP 160$
1051 014104 10$:
1052 014104 004737 024302 JSR PC,SCTCMP ;FORCE SECTOR COMPARE USING SUBROUTINE
```

```

014110 000402 BR 20$ ;BRANCH TO 20$ IF NO ERROR
014112 104000 EMT
1053 014114 000565 BR 160$
1054
1055 ;WRITE PROM OF DATA TIMING SEQUENCER SHOULD NOW BE ENABLED
1056 ;FIRST, VERIFY THAT WRITE GATE IS ON, INDICATING THAT
1057 ;SECTOR COMPARE SET
1058 014116 20$:
1059 014116 016037 000040 001142 MOV RMMR2(R0), $BDDAT ;STORE RMMR2 AT $BDDAT
1060 014124 042737 177776 001142 BIC #^CBB00, $BDDAT ;BUS BIT 0 IS WRITE GATE
1061 014132 001011 BNE 30$ ;
1062 014134 005037 001140 CLR $GDDAT
1063 014140 010037 001136 MOV R0, $BDADR
1064 014144 062737 000040 001136 ADD #RMMR2, $BDADR
1065 014152 104042 EMT 42
1066 014154 000545 BR 160$ ;FORMAT-CS FAILURE
1067
1068 ;STEP THE DATA TIMING SEQUENCER TO LOCATION 16 AND VERIFY WRITE
1069 ;GATE STAYS ON
1070 014156 30$:
1071 014156 012702 000016 MOV #14., R2 ;MAXMUM NUMBER OF CLOCK
1072 014162 40$:
1073 014162 016037 000040 001142 MOV RMMR2(R0), $BDDAT ;STORE RMMR2 AT $BDDAT
1074 014170 032737 000001 001142 BIT #BB00, $BDDAT ;
1075 014176 001014 BNE 50$ ;BRANCH IF WRITE GATE ON
1076 014200 005302 DEC R2
1077 014202 001367 BNE 40$
1078 014204 010037 001136 MOV R0, $BDADR
1079 014210 062737 000040 001136 ADD #RMMR2, $BDADR
1080 014216 012737 000001 001140 MOV #BB00, $GDDAT
1081 014224 104042 EMT 42
1082 014226 000520 BR 160$ ;FORMAT - CS FAILURE
1083 014230 50$:
1084
1085 014230 60$:
1086 014230 012760 055401 000024 MOV #MR1AAA!MCLK, RMMR1(R0) ;LOAD RMMR1
1087 014236 012760 051401 000024 MOV #MR1AAA, RMMR1(R0) ;LOAD RMMR1
1088 014244 016003 000024 MOV RMMR1(R0), R3 ;STORE RMMR1 AT R3
1089 014250 032703 000040 BIT #WC, R3
1090 014254 001765 BEQ 60$
1091
1092 ;PROM STROBE CAME ON-DECREMENT PROM STROBE COUNT
1093 014256 005302 DEC R2
1094 014260 001414 BEQ 80$
1095
1096 ;WAIT FOR PROM STROBE TO GO OFF, THEN REPEAT LOOP
1097 014262 70$:
1098 014262 012760 055401 000024 MOV #MR1AAA!MCLK, RMMR1(R0) ;LOAD RMMR1
1099 014270 012760 051401 000024 MOV #MR1AAA, RMMR1(R0) ;LOAD RMMR1
1100 014276 016003 000024 MOV RMMR1(R0), R3 ;STORE RMMR1 AT R3
1101 014302 032703 000040 BIT #WC, R3
1102 014306 001725 BEQ 40$
1103 014310 000764 BR 70$
1104
1105 ;VERIFY HEADER SYNC GENERATION
1106 ;WRITE DATA BIT IS INVERTED AT MAINTENANCE REGISTER
1107 ;FIRST, COUNT NUMBER OF ZERO BITS UNTIL FIRST ONE BIT

```

```

1108 014312      80$:
1109 014312 012702 000020      MOV    #16.,R2      ;MAX TIMES THRU LOOP
1110 014316 005003      CLR    R3
1111 014320      90$:
      MOV    RMMR1(R0),R4      ;STORE RMMR1 AT R4
      BIT    #MWD,R4
1112 014324 016004 000024      BEQ    110$      ;JUMP OUT OF LOOP WITH FIRST 1
1113 014330 001414      INC    R3      ;INCREMENT ZERO COUNT
1114 014332 005203      DEC    R2      ;DECREMENT MAX LOOP COUNT
1115 014334 005302      BNE    100$
1116 014336 001002      EMT    43
1117 014340 104043      BR     160$      ;HEADER SYNC ,CANT GET FIRST 1
1118 014342 000452
1119 014344      100$:
1120 014344 012760 055401 000024      MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1121 014352 012760 051401 000024      MOV    #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1122 014360 000757      BR     90$
1123
1124      ;MAKE SURE THERE WERE AT LEAST 8 ZERO BITS IN HEADER SYNC
1125 014362      110$:
1126 014362 020327 000010      CMP    R3,#8.
1127 014366 103002      BHIS  120$
1128 014370 104043      EMT    43
1129 014372 000436      BR     160$      ;HEADER SYNC
1130
1131      ;SAMPLE AND STORE THE REST OF THE HEADER SYNC
1132 014374      120$:
1133 014374 012702 000010      MOV    #8.,R2      ;NUMBER OF SAMPLES
1134 014400 005003      CLR    R3      ;HEADER SYNC
1135 014402      130$:
1136 014402 016004 000024      MOV    RMMR1(R0),R4      ;STORE RMMR1 AT R4
1137 014406 006303      ASL   R3
1138 014410 032704 000010      BIT    #MWD,R4      ;SHIFT SAMPLE,IF SYNC BIT IS 1
1139 014414 001002      BNE    140$      ;THEN SET SAMPLE INPUT
1140 014416 052703 000001      BIS    #BIT0,R3
1141 014422 005302      140$:
1142 014424 001407      DEC    R2
1143 014426 012760 055401 000024      BEQ    150$
1144 014434 012760 051401 000024      MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1145 014442 000757      MOV    #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1146      BR     130$      ;DO LOOP TIL R2=0
1147
1148 014444      ;VERIFY THE SYNC BIT STREAM IS 000000010011000
1149 014444 012737 000230 001140      150$:
1150 014452 010337 001142      MOV    #000230,$GDDAT
1151 014456 023737 001140 001142      MOV    R3,$BDDAT
1152 014464 001401      CMP    $GDDAT,$BDDAT
1153 014466 104044      BEQ    160$
1154 014470      EMT    44
1155
1156      ;*****
      ;*TEST 21      WRITE HEADER TEST
      ;*****
TST21:
014470      SCOPE      ;SCOPE CALL
014470 000004      NOP
014472 000240      MOV    #STACK,SP      ;LOAD THE STACK POINTER
014474 012706 001100

```

```

014500 013700 001276          MOV    $BASE,R0          ;R0 = UNIBUS ADDRESS
014504 013701 001462          MOV    TSTQUE,R1        ;R1 = POINTER TO DEVICE
014510 012737 000021 001226  MOV    #21,$TESTN      ;:SET TEST NUMBER IN APT MAIL BOX

1157
1158          ;:SETUP THE REGISTER OUTPUT BUFFER FOR SUBROUTINES
1159 014516 012737 001466 001442  MOV    #822.,RMDCO      ;LAST CYLINDER
1160 014524 013737 001330 001414  MOV    LSTRK,RMDAO      ;SET LAST TRACK AND
1161 014532 112737 000036 001414  MOV    #30.,RMDAO      ;LAST SECTOR
1162 014540 012737 010000 001440  MOV    #FMT16,RMOFO    ;IN 16 BIT FORMAT
1163 014546 012702 054244          MOV    #BUFFER,R2      ;BUFFER ADDRESS
1164 014552 010237 001412          MOV    R2,RMBAO        ;BUFFER STARTS
1165          ;STORE FIRST HEADER WORD
1166 014556 012712 150000          MOV    #150000,(R2)     ;SET MF/UF BITS GOOD, 16 BIT FORMAT AND
1167 014562 052722 001466          BIS    #822.,(R2)+    ;LAST CYLINDER
1168          ;STORE SECOND HEADER WORD
1169 014566 013712 001330          MOV    LSTRK,(R2)     ;SET LAST TRACK ADDRESS AND
1170 014572 112712 000036          MOV    #30.,(R2)     ;SECTOR ADDRESS.
1171 014576 012737 177776 001410  MOV    #-2,RMWCO       ;2 WORDS (2'S COMP)
1172 014604 012737 000063 001406  MOV    #WH!GO,RMCS10   ;WRITE HEADER COMMAND
1173
1174 014612 004737 023430          JSR    PC,ENBSCH       ;EXECUTE DATA COMMAND TO POINT WHERE
                                ;SEARCH IS ENABLED USING SUBROUTINE.
                                ;BRANCH TO 10$ IF NO ERROR
                                10$:
                                BR    10$
                                EMT
                                JMP    160$
1175 014622 000137 015456          10$:
1176 014626          JSR    PC,SCTCMP       ;FORCE SECTOR COMPARE USING SUBROUTINE
1177 014626 004737 024302          BR    20$             ;BRANCH TO 20$ IF NO ERROR
                                20$:
                                EMT
                                JMP    160$
1178 014632 000403          ;STEP THE DATA TIMING SEQUENCER UNTIL 'HEADER AREA' COMES ON
1179 014636 000137 015456          20$:
1180          MOV    #15.,R2          ;ALLOW 15 MORE PROM STORBES
1181 014642          MOV    #33.,R4          ;ALLOW 33 BIT CLOCKS PER PROM STROBE
1182 014642 012702 000017          30$:
1183 014646 012704 000041          MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1184          MOV    #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1185          MOV    RMMR1(R0),R3     ;STORE RMMR1 AT R3
1186 014652          BIT    #WC,R3
1187 014652 012760 055401 000024  BNE    50$
1188 014660 012760 051401 000024  DEC    R4
1189 014666 016003 000024          BNE    40$
1190 014672 032703 000040          JMP    160$           ;COUNT EXHAUSTED
1191 014676 001004          ;SEE IF HEADER AREA CAME ON
1192 014700 005304          50$:
1193 014702 001363          BIT    #PHA,R3
1194 014704 000137 015456          BNE    80$
1195          DEC    R2
1196          BNE    60$
1197 014710          ;ERROR-HEADER AREA NEVER CAME ON
1198 014710 032703 000200          MOV    #PHA,$GDDAT     ;SETUP ERROR MESSAGE
1199 014714 001062          CLR    $BDDAT
1200 014716 005302
1201 014720 001015
1202
1203
1204 014722 012737 000200 001140
1205 014730 005037 001142
    
```

```
1206 014734 010037 001136          MOV    R0,$BDADR
1207 014740 062737 000024 001136  ADD    #RMMR1,$BDADR
1208 014746 104045          EMT    45
1209 014750 000137 015456          JMP    160$           ;SKIP REST OF TEST
1210
1211          ;WAIT FOR PROM STROBE TO GO OFF
1212 014754          60$:
1213 014754 012760 055401 000024  MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1214 014762 012760 051401 000024  MOV    #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1215 014770 016003 000024          MOV    RMMR1(R0),R3          ;STORE RMMR1 AT R3
1216 014774 032703 000040          BIT    #WC,R3
1217 015000 001404          BEQ    70$
1218 015002 005304          DEC    R4
1219 015004 001363          BNE    60$
1220 015006 000137 015456          JMP    160$           ;COUNT EXHAUSTED
1221
1222          ;VERIFY THE TAG BUS ONCE EVERY PROM STROBE
1223 015012          70$:
1224 015012 016037 000040 001142  MOV    RMMR2(R0),$BDDAT      ;STORE RMMR2 AT $BDDAT
1225 015020 042737 176000 001142  BIC    #^C1777,$BDDAT
1226 015026 022737 000001 001142  CMP    #BB00,$BDDAT          ;WRITE GATE SHOULD BE ON
1227 015034 001704          BEQ    30$              ;ALL ELSE OFF
1228 015036 010037 001136          MOV    R0,$BDADR
1229 015042 062737 000040 001136  ADD    #RMMR2,$BDADR
1230 015050 012737 000001 001140  MOV    #BB00,$GDDAT
1231 015056 104042          EMT    42
1232 015060 000576          BR    160$
1233
1234          ;HEADER AREA IS NOW ON CAN START SAMPLE
1235          ;DATA AFTER 5 MORE BIT CLOCKS
1236 015062          80$:
1237 015062 012702 000005          MOV    #5,R2
1238 015066          81$:
1239 015066 012760 055401 000024  MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1240 015074 012760 051401 000024  MOV    #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1241 015104 005302          DEC    R2
1242          BNE    81$
1243          ;SAMPLE AND STORE THE TWO HEADER WORDS
1244 015106 005037 001174          CLR    $TMP0
1245 015112 005037 001176          CLR    $TMP1
1246 015116 012702 001174          MOV    #$TMP0,R2             ;R2 NUMBER OF WORDS/ADDRESS
1247 015122 012703 000020          90$: MOV    #16.,R3              ;NUMBER OF BITS
1248 015126 005004          CLR    R4                   ;WORD STORAGE
1249 015130          100$:
1250 015130 016005 000024          MOV    RMMR1(R0),R5          ;STORE RMMR1 AT R5
1251 015134 006304          ASL    R4                   ;SHIFT WORD SAMPLE
1252 015136 032705 000010          BIT    #MWD,R5              ;SET BIT 0 IF WRITE BIT ON
1253 015142 001002          BNE    110$
1254 015144 052704 000001          BIS    #BIT0,R4
1255 015150          110$:
1256 015150 012760 055401 000024  MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1257 015156 012760 051401 000024  MOV    #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1258 015166 005303          DEC    R3
1259 015170 010422          BNE    100$
1260 015172 022702 001200          MOV    R4,(R2)+             ;STORE WORD SAMPLE
          CMP    #$TMP0+4,R2      ;DONE ?
```

```

1261 015176 101415          BLOS      120$          ;YES
1262
1263          ;HEADER AREA SHOULD STILL BE ON FOR SECOND HEADER WORD
1264 015200 042705 177577    BIC      #^C<PHA>,R5
1265 015204 001346          BNE      90$          ;BRANCH IF ON
1266 015206 010037 001136    MOV      R0,$BDADR
1267 015212 062737 000024 001136    ADD      #RMMR1,$BDADR
1268 015220 012737 000200 001140    MOV      #PHA,$GDDAT
1269 015226 104045          EMT      45
1270 015230 000512          BR       160$
1271
1272          ;WAIT UNTIL ECRC SETS
1273 015232          120$:
1274 015232 005003          CLR      R3          ;CHECK NUMBER BITS DIFFER BET PHA AND ECRC
1275
1276 015234          125$:
1277 015234 016005 000024    MOV      RMMR1(R0),R5 ;STORE RMMR1 AT R5
1278 015244 001026          BIC      #^C<ECRC>,R5 ;CHECK OUT THE REST BITS
1279 015246 012760 055401 000024    BNE      127$        ;BRANCH IF ECRC ON
1280 015254 012760 051401 000024    MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1281 015262 005203          MOV      #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1282 015264 022703 000020    INC      R3          ;TOTAL NUMBER OF BIT DIFF
1283 015270 101361          CMP      #16.,R3     ;OVER ONE WORD ?
1284 015272 010037 001136    BHI      125$        ;BRANCH IF NOT
1285 015276 062737 000024 001136    MOV      R0,$BDADR
1286 015304 012737 001000 001140    ADD      #RMMR1,$BDADR
1287 015312 005037 001142    MOV      #ECRC,$GDDAT
1288 015316 104046          CLR      $BDDAT     ;CLEAR THE EXP
1289 015320 000456          EMT      46
1290          BR       160$          ;EXIT
1291
1292          ;SAMPLE AND STORE CRC WORD
1293 015322 012703 000020    127$:
1294 015326 005004          MOV      #16.,R3
1295 015330          CLR      R4
1296 015330 016005 000024    130$:
1297 015334 006304          MOV      RMMR1(R0),R5 ;STORE RMMR1 AT R5
1298 015336 032705 000010    ASL      R4          ;SHIFT WORD SAMPLE
1299 015342 001002          BIT      #MWD,R5     ;SET BIT 0 IF WRITE BIT ON
1300 015350          BNE      140$
1301 015350 012760 055401 000024    BIS      #BIT0,R4
1302 015356 012760 051401 000024    MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1303 015364 005303          MOV      #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1304 015370 010422          DEC      R3
1305          BNE      130$
1306          MOV      R4,(R2)+
1307
1308          ;VERIFY 3 HEADER WORDS
1309          ;NOTE THAT DATA WAS STORED IN THE REVERSE
1310          ;ORDER FROM WHICH IT WAS WRITTEN
1311 015372 022737 066313 001174    CMP      #066313,$TMP0 ;CORRECT FIRST HDR WRD ?
1312 015400 001025          BNE      150$        ;NO !!
1313
1314          ;VERIFY 2ND AND 3RD HEADER WORDS FOR AN RM05
1315 015402 032737 010000 001330    BIT      #TA16,LSTRK ;IS DEVICE AN RM05 ?
1316 015410 001411          BEQ     145$        ;NO !!

```

```

1315 015412 022,37 074110 001176      CMP      #074110,$STMP1      ;CORRECT SECOND HDR WRD ?
1316 015420 001015                      BNE      150$               ;NO !!
1317 015422 022737 167073 001200      CMP      #167073,$STMP2     ;CORRECT HEADER CRC ?
1318 015430 001011                      BNE      150$               ;NO !!
1319 015432 000411                      BR       160$
1320
1321                                     ;VERIFY 2ND AND 3RD HEADER WORDS FOR AN RM03
1322 015434                                     145$:
1323 015434 022737 074040 001176      CMP      #074040,$STMP1     ;CORRECT SECOND HDR WRD ?
1324 015442 001004                      BNE      150$               ;NO !!
1325 015444 022737 067510 001200      CMP      #067510,$STMP2     ;CORRECT HEADER CRC ?
1326 015452 001401                      BEQ      160$               ;YES !!
1327 015454                                     150$:
1328 015454 104047                      EMT      47
1329 015456                                     160$:                               ;END OF SUB TEST
1330
1331                                     ;*****
;*TEST 22          HEADER COMPARE TEST
;*****
TST22:
015456                                     SCOPE                               ;SCOPE CALL
015456 000004                                     NOP
015460 000240                                     MOV      #STACK,SP           ;LOAD THE STACK POINTER
015462 012706 001100                                     MOV      $BASE,R0           ;R0 = UNIBUS ADDRESS
015466 013700 001276                                     MOV      TSTQUE,R1          ;R1 = POINTER TO DEVICE
015472 013701 001462                                     MOV      #22,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
015476 012737 000022 001226
1332
1333                                     ;LOAD REGISTER OUTPUT BUFFER FOR READ HEADER COMMAND
1334 015504 012737 001466 001442      MOV      #822.,RMDCO        ;LAST CYLINDER
1335 015512 013737 001330 001414      MOV      LSTRK,RMDAO        ;SET LAST TRACK AND
1336 015520 112737 000036 001414      MOVB     #30.,RMDAO         ;LAST SECTOR
1337 015526 012737 054244 001412      MOV      #BUFFER,RMBAO      ;DATA ADDRESS
1338 015534 012737 177776 001410      MOV      #-2,RMWCO          ;2 WORDS (2'S COMP)
1339 015542 012737 010000 001440      MOV      #FMT16,RMOFO       ;16 BIT MODE
1340 015550 012737 000073 001406      MOV      #RH!GO,RMCS10      ;READ HEADER AND DATA FUN
1341
1342 015556 004737 023430              JSR      PC,ENBSCH           ;EXECUTE DATA COMMAND TO POINT WHERE
;SEARCH IS ENABLED USING SUBROUTINE.
015562 000403                          BR       10$                 ;BRANCH TO 10$ IF NO ERROR
015564 104000                          EMT
1343 015566 000137 016724              JMP      230$                ;SKIP REST OF TEST IF ERROR
1344 015572                                     10$:
1345 015572 004737 024302              JSR      PC,SCTCMP          ;FORCE SECTOR COMPARE USING SUBROUTINE
015576 000403                          BR       20$                 ;BRANCH TO 20$ IF NO ERROR
015600 104000                          EMT
1346 015602 000137 016724              JMP      230$
1347
1348                                     ;READ GATE SHOULD BE OFF FOR 3 PROM STROBES
1349 015606                                     20$:
1350 015606 010037 001136 001136      MOV      R0,$BDADR          ;SETUP ERROR MESSAGE
1351 015612 062737 000040                      ADD      #RMMR2,$BDADR
1352 015620 012702 000004                      MOV      #4,R2              ;R2=NUMBER OF PROM STROBES
1353 015624 012704 000021                      MOV      #17.,R4           ;MAX BIT CLOCKS
1354
1355                                     ;CLOCK BIT CLOCK UNTIL PROM STROBE COMES ON
1356 015630                                     40$:

```

```
1357 015630 012760 055401 000024      MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1358 015636 012760 051401 000024      MOV      #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1359 015644 016003 000024      MOV      RMMR1(R0),R3           ;STORE RMMR1 AT R3
1360 015650 032703 000040      BIT      #WC,R3
1361 015654 001004      BNE      50$                   ;BRANCH WHEN PROM STROBE ON
1362 015656 005304      DEC      R4
1363 015660 001363      BNE      40$
1364 015662 000137 016724      JMP      230$
1365
1366      ;VERIFY READ GATE IS OFF AND TAG BUS IS ZERO
1367 015666 50$:
1368 015666 016003 000040      MOV      RMMR2(R0),R3           ;STORE RMMR2 AT R3
1369 015672 042703 176000      BIC      #^C1777,R3
1370 015676 001407      BEQ      60$                   ;BRANCH IF TAG BUS ZERO
1371 015700 010337 001142      MOV      R3,$BDDAT
1372 015704 005037 001140      CLR      $GDDAT
1373 015710 104050      EMT      50
1374 015712 000137 016724      JMP      230$                   ;READ HEADER FUNCTION
1375
1376      ;CLOCK BIT CLOCK TIL PROM STROBE GOES OFF
1377 015716 60$:
1378 015716 012760 055401 000024      MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1379 015724 012760 051401 000024      MOV      #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1380 015732 016003 000024      MOV      RMMR1(R0),R3           ;STORE RMMR1 AT R3
1381 015736 032703 000040      BIT      #WC,R3
1382 015742 001404      BEQ      70$                   ;BRANCH IF PROM STROBE OFF
1383 015744 005304      DEC      R4
1384 015746 001363      BNE      60$
1385 015750 000137 016724      JMP      230$
1386
1387      ;CONTINUE FOR 3 BIT CYCLE TOTAL (LOC 0 NOT SAMPLED)
1388 015754 70$:
1389 015754 005302      DEC      R2
1390 015756 001322      BNE      30$
1391
1392      ;READ GATE SHOULD COME ON WITH NEXT PROM STROBE AND STAY
1393      ;ON FOR 7 CYCLES (AND MORE)
1394
1395 015760 012702 000006      MOV      #6,R2
1396 015764 012703 000021 80$:      MOV      #17.,R3
1397
1398      ;CLOCK BIT CLOCK TIL PROM STROBE SETS
1399 015770 90$:
1400 015770 012760 055401 000024      MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1401 015776 012760 051401 000024      MOV      #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1402 016004 016004 000024      MOV      RMMR1(R0),R4           ;STORE RMMR1 AT R4
1403 016010 032704 000040      BIT      #WC,R4
1404 016014 001004      BNE      100$                  ;BRANCH WHEN PROM STROBE ON
1405 016016 005303      DEC      R3
1406 016020 001363      BNE      90$
1407 016022 000137 016724      JMP      230$
1408
1409      ;VERIFY READ GATE ON AND REST OF TAG BUS OFF
1410 016026 100$:
1411 016026 016004 000040      MOV      RMMR2(R0),R4           ;STORE RMMR2 AT R4
1412 016032 042704 176000      BIC      #^C1777,R4
1413 016036 022704 000002      CMP      #BB01,R4
```



```

1414 016042 001410          BEQ      110$          ;BRANCH IF READ GATE ON
1415 016044 010437 001142          MOV      R4,$BDDAT
1416 016050 012737 000002 001140          MOV      #8801,$GDDAT
1417 016056 104050          EMT      50
1418 016060 000137 016724          JMP      230$          ;READ HEADER FUNCTION
1419
1420          ;CLOCK BIT CLOCK TIL PROM STROBE GOES OFF
1421 016064          110$:
1422 016064 012760 055401 000024          MOV      #MMR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1423 016072 012760 051401 000024          MOV      #MMR1AAA,RMMR1(R0)      ;LOAD RMMR1
1424 016100 016004 000024          MOV      RMMR1(R0),R4            ;STORE RMMR1 AT R4
1425 016104 032704 000040          BIT      #WC,R4
1426 016110 001404          BEQ      120$          ;BRANCH WHEN PROM STROBE OFF
1427 016112 005303          DEC      R3
1428 016114 001363          BNE      110$
1429 016116 000137 016724          JMP      230$
1430
1431          ;CONTINUE FOR TOTAL OF 7 CYCLES
1432 016122          120$:
1433 016122 005302          DEC      R2
1434 016124 001317          BNE      80$
1435
1436          ;LOOKING FOR SYNC SHOULD SET WITH NEXT PROM STORBE
1437 016126 012702 000021          MOV      #17.,R2                ;MAX BIT CLOCKS
1438
1439          ;CLOCK BIT CLOCK TIL PROM STROBE SETS
1440 016132          130$:
1441 016132 012760 055401 000024          MOV      #MMR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1442 016140 012760 051401 000024          MOV      #MMR1AAA,RMMR1(R0)      ;LOAD RMMR1
1443 016146 016003 000024          MOV      RMMR1(R0),R3            ;STORE RMMR1 AT R3
1444 016152 032703 000040          BIT      #WC,R3
1445 016156 001004          BNE      140$          ;BRANCH WHEN PROM STROBE ON
1446 016160 005302          DEC      R2
1447 016162 001363          BNE      130$
1448 016164 000137 016724          JMP      230$
1449
1450          ;VERIFY 'PLFS' IS NOW ON
1451 016170          140$:
1452 016170 042703 175777          BIC      #^CPLFS,R3
1453 016174 001015          BNE      150$
1454 016176 010337 001142          MOV      R3,$BDDAT
1455 016202 012737 002000 001140          MOV      #PLFS,$GDDAT
1456 016210 010037 001136          MOV      R0,$BDADR
1457 016214 062737 000024 001136          ADD      #RMMR1,$BDADR
1458 016222 104035          EMT      35
1459 016224 000137 016724          JMP      230$
1460 016230          150$:
1461 016230 012703 000005          MOV      #5,R3
1462
1463          ;WITH PLFS ON, WORD COUNT INHIBIT WILL SET IN 5 BIT CLOCKS
1464 016234          155$:
1465 016234 012760 055401 000024          MOV      #MMR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1466 016242 012760 051401 000024          MOV      #MMR1AAA,RMMR1(R0)      ;LOAD RMMR1
1467 016250 005303          DEC      R3
1468 016252 001370          BNE      155$
1469
1470          ;SIMULATE THE SYNC PATTERN BEING READ

```

```

1471 016254 012702 000031      MOV      #31,R2      ;SYNC PATTERN=00011001
1472 016260 012703 000010      MOV      #8.,R3      ;8 BITS IN PATTERN
1473 016264      160$:
1474 016264 012737 055401 001432      MOV      #MR1AAA!MCLK,RMMR10
1475 016272 000241      CLC
1476 016274 006002      ROR      R2
1477 016276 103003      BCC      165$
1478 016300 052737 002000 001432      BIS      #MRD,RMMR10
1479 016306      165$:
1480 016306 013760 001432 000024      MOV      RMMR10,RMMR1(R0)      ;LOAD RMMR1
1481 016314 042737 004000 001432      BIC      #MCLK,RMMR10
1482 016322 013760 001432 000024      MOV      RMMR10,RMMR1(R0)      ;LOAD RMMR1
1483 016330 005303      DEC      R3      ;CONTINUE TIL SHIFT COUNT
1484 016332 001354      BNE      160$      ;IS ZERO
1485
1486      ;SIMULATE THE HEADER BEING READ
1487 016334 012702 001174      MOV      #STMP0,R2
1488 016340 012737 150000 001174      MOV      #150000,$TMP0      ;SET MF/UF BITS GOOD, 16 BIT FORMAT AND
1489 016346 052737 001466 001174      BIS      #822.,$TMP0      ;LAST CYLINDER
1490 016354 013737 001350 001176      MOV      LSTRK,$TMP1      ;SET LAST TRACK AND
1491 016362 112737 000036 001176      MOV      #30.,$TMP1      ;LAST SECTOR
1492 016370 012737 011366 001200      MOV      #011366,$TMP2      ;GET CRC PATTERN
1493 016376 032737 010000 001330      BIT      #TA16,LSTRK      ;IS DEVICE AN RM05 ?
1494 016404 001403      BEQ      170$      ;NO !!
1495 016406 012737 156167 001200      MOV      #156167,$TMP2      ;YES, ADJUST CRC PATTERN
1496 016414      170$:
1497 016414 012703 000020      MOV      #16.,R3      ;NUMBER OF BITS EACH WORD
1498 016420 012204      MOV      (R2)+,R4      ;HEADER WORD 1,2 OR 3
1499 016422      175$:
1500 016422 012737 055401 001432      MOV      #MR1AAA!MCLK,RMMR10
1501 016430 000241      CLC
1502 016432 006004      ROR      R4
1503 016434 103003      BCC      180$
1504 016436 052737 002000 001432      BIS      #MRD,RMMR10
1505 016444      180$:
1506 016444 013760 001432 000024      MOV      RMMR10,RMMR1(R0)      ;LOAD RMMR1
1507 016452 042737 004000 001432      BIC      #MCLK,RMMR10
1508 016460 013760 001432 000024      MOV      RMMR10,RMMR1(R0)      ;LOAD RMMR1
1509 016466 005303      DEC      R3      ;SHIFT OUT 16 BITS
1510 016470 001354      BNE      175$
1511 016472 020227 001200      CMP      R2,#STMP0+4      ;ALL DONE ?
1512 016476 101746      BLOS     170$      ;BRANCH IF NOT
1513
1514 016500 012702 000006      ;LOOKING FOR SYNC SHOULD COME ON WITHIN 6 STROBES
1515 016504 012703 000021      185$: MOV      #6,R2
1516      MOV      #17.,R3
1517      ;CLOCK UNTIL PROM STROBE ON
1518 016510      190$:
1519 016510 012760 055401 000024      MOV      #MR1AAA!MCLK,RMMR1(R0)      ;LOAD RMMR1
1520 016516 012760 051401 000024      MOV      #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1521 016524 016004 000024      MOV      RMMR1(R0),R4      ;STORE RMMR1 AT R4
1522 016530 032704 000040      BIT      #WC,R4
1523 016534 001003      BNE      195$
1524 016536 005303      DEC      R3
1525 016540 001363      BNE      190$
1526 016542 000470      BR      230$

```

```

1527
1528
1529 016544
1530 016544 042704 175777
1531 016550 001034
1532
1533
1534 016552 005302
1535 016554 001014
1536 016556 010437 001142
1537 016562 012737 002000 001140
1538 016570 010037 001136
1539 016574 062737 000024 001136
1540 016602 104035
1541 016604 000447
1542
1543
1544 016606
1545 016606 012760 055401 000024
1546 016614 012760 051401 000024
1547 016622 016004 000024
1548 016626 032704 000040
1549 016632 001724
1550 016634 005303
1551 016636 001363
1552 016640 000431
1553
1554
1555 016642
1556 016642 016037 000014 001142
1557 016650 042737 177157 001142
1558 016656 001411
1559 016660 005037 001140
1560 016664 010037 001136
1561 016670 062737 000014 001136
1562 016676 104051
1563 016700 000411
1564
1565
1566 016702
1567 016702 023737 054244 001174
1568 016710 001004
1569 016712 023737 054246 001176
1570 016720 001401
1571
1572
1573 016722
1574 016722 104052
1575 016724
1576
1577

016724
016724 000004
016726 000240

;SEE IF 'PLFS' IS ON
195$:
    BIC    #^CPLFS,R4
    BNE    210$

;CONTINUE IF LESS THAN 6 PROM STROBES
    DEC    R2
    BNE    200$
    MOV    R4,$BDDAT    ;SETUP ERROR
    MOV    #PLFS,$GDDAT
    MOV    R0,$BDADR
    ADD    #RMMR1,$BDADR
    EMT    35
    BR     230$    ;HEADER

;CLOCK UNTIL PROM STROBE OFF
200$:
    MOV    #RMR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
    MOV    #RMR1AAA,RMMR1(R0) ;LOAD RMMR1
    MOV    RMMR1(R0),R4 ;STORE RMMR1 AT R4
    BIT    #WC,R4
    BEQ    185$
    DEC    R3
    BNE    200$
    BR     230$

;VERIFY NO HEADER ERROR IS SET
210$:
    MOV    RMER1(R0),$BDDAT ;STORE RMER1 AT $BDDAT
    BIC    #^C<HCRC!HCE!FER>,$BDDAT
    BEQ    220$
    CLR    $GDDAT
    MOV    R0,$BDADR
    ADD    #RMER1,$BDADR
    EMT    51
    BR     230$    ;HEADER ERROR SET

;VERIFY DATA IN MEMORY OK
220$:
    CMP    BUFFER,$TMP0 ;COMPARE 1ST HEADER WORD
    BNE    225$
    CMP    BUFFER+2,$TMP1 ;COMPARE 2ND HEADER WORD
    BEQ    230$

;REPORT ERROR IN ONE OR MORE HEADER WORDS
225$:
    EMT    52
230$:

;*****
;*TEST 23      ECC GENERATION TEST
;*****

TST23:
    SCOPE
    NOP
;SCOPE CALL
  
```

```

016730 012706 001100      MOV    #STACK,SP      ;LOAD THE STACK POINTER
016734 013700 001276      MOV    $BASE,R0       ;R0 = UNIBUS ADDRESS
016740 013701 001462      MOV    TSTQUE,R1      ;R1 = POINTER TO DEVICE
016744 012737 000023 001226  MOV    #23,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

1578
1579      ;SETUP REGISTER OUTPUT BUFFER FOR WRITE DATA COMMAND
1580 016752 012737 001466 001442  MOV    #822.,RMDCO     ;LAST CYLINDER
1581 016760 013737 001330 001414  MOV    LSTRK,RMDAO     ;SET LAST TRACK AND
1582 016766 112737 000036 001414  MOV    #30.,RMDAO     ;LAST SECTOR
1583 016774 012737 012000 001440  MOV    #FMT16!HCI,RMOFO ;16 BIT FORMAT,IGNORE HEADER
1584 017002 012702 054244      MOV    #BUFFER,R2
1585 017006 010237 001412      MOV    R2,RMBAO       ;DATA ADDRESS
1586 017012 012703 177400      MOV    #-256.,R3      ;256. WORDS (2'S COMP)
1587 017016 010337 001410      MOV    R3,RMWCO
1588 017022 012737 000061 001406  MOV    #WD!GO,RMCS10  ;WRITE DATA COMMAND
1589
1590      ;FILL THE DATA BUFFER WITH ALL ONES
1591 017030 012704 177777      MOV    #177777,R4     ;DATA PATTERN ALL ONES
1592 017034 012703 000400      MOV    #256.,R3      ;ONE SECTOR DATA FIELD
1593 017040 010422 10$:      MOV    R4,(R2)+
1594 017042 005303      DEC    R3
1595 017044 001375      BNE    10$
1596
1597 017046 004737 023430      JSR    PC,ENBSCH      ;EXECUTE DATA COMMAND TO POINT WHERE
                        ;SEARCH IS ENABLED USING SUBROUTINE.
                        ;BRANCH TO 20$ IF NO ERROR
                        017052 000403      BR     20$
                        017054 104000      EMT
1598 017056 000137 020112      JMP    400$
1599 017062 20$:
1600 017062 004737 024302      JSR    PC,SCTCMP     ;FORCE SECTOR COMPARE USING SUBROUTINE
                        BR     30$          ;BRANCH TO 30$ IF NO ERROR
                        017066 000403      EMT
                        017070 104000      JMP    400$
1601 017072 000137 020112 30$:
1602 017076 30$:
1603 017076 004737 024410      JSR    PC,SETLFS     ;SET 'LOOKING FOR SYNC' USING SUBROUTINE
                        BR     40$          ;BRANCH TO 40$ IF NO ERROR
                        017102 000403      EMT
                        017104 104000      JMP    400$
1604 017106 000137 020112 40$:
1605 017112 40$:
1606 017112 004737 024572      JSR    PC,CLKSNC     ;CLOCK THE SYNC PATTERN USING SUBROUTINE
                        BR     50$          ;BR TO 50$ IF NO ERROR
                        017116 000403      EMT
                        017120 104000      JMP    400$
1607 017122 000137 020112
1608
1609      ;HEADER COMPARE IS INHIBITED FOR THIS TEST
1610      ;CLOCK THE DATA TIMING SEQUENCER AND VERIFY THE FOLLOWING WAVEFORMS
1611      ;FOR EACH LEADING EDGD OF PROM STROBE
1612      ;   HEADER ARES, READ GATE (DATA TIMING SEQUENCER=200 OCTAL)
1613      ;   ENABLE CRC OUT
1614      ;   WRITE GATE
1615
1616 017126 012702 017660 50$:  MOV    #230$,R2      ;POINTER TO WAVE FORM TABLE
1617
1618      ;CLOCK BIT CLOCK (MCLK) UNTIL PROM STROBE COMES ON
1619 017132 70$:
1620 017132 004737 017700      JSR    PC,300$
1621

```

```

1622      ;PROM STROBE IS ON - VERIFY WAVEFORM
1623 017136 80$:
1624 017136 016037 000024 001142      MOV      RMMR1(R0), $BDDAT      ;STORE RMMR1 AT $BDDAT
1625 017144 042737 176157 001142      BIC      #^C<ECRC!PDA!PHA!EECL>, $BDDAT
1626 017152 012237 001140      MOV      (R2)+, $GDDAT
1627 017156 023737 001140 001142      CMP      $GDDAT, $BDDAT
1628 017164 001410      BEQ      90$      ;BRANCH IF RMMR1 OK
1629
1630      ;ERROR - THE DATA TIMING SEQUENCER OUTPUT IS ONCORRECT
1631 017166 010037 001136      MOV      R0, $BDADR
1632 017172 062737 000024 001136      ADD      #RMMR1, $BDADR
1633 017200 104054      EMT      54
1634 017202 000137 020112      JMP      400$
1635
1636      ;VERIFY THE TAB BUS
1637 017206 90$:
1638 017206 016037 000040 001142      MOV      RMMR2(R0), $BDDAT      ;STORE RMMR2 AT $BDDAT
1639 017214 042737 176000 001142      BIC      #^C1777, $BDDAT
1640 017222 012237 001140      MOV      (R2)+, $GDDAT
1641 017226 023737 001140 001142      CMP      $GDDAT, $BDDAT
1642 017234 001410      BEQ      100$      ;BRANCH IF TAG BUS OK
1643
1644      ;ERROR- TAG BUS IS WRONG
1645 017236 010037 001136      MOV      R0, $BDADR
1646 017242 062737 000040 001136      ADD      #RMMR2, $BDADR
1647 017250 104053      EMT      53
1648 017252 000137 020112      JMP      400$
1649
1650      ;CLOCK BIT CLOCK TIL PROM STROBE RESETS
1651 017256 100$:
1652 017256 004737 020006      JSR      PC, 350$
1653
1654      ;CONTINUE CHECKING THROUGH 4 PROM CYCLES
1655 017262 110$:
1656 017262 020227 017676      CMP      R2, #230$+16
1657 017266 103721      BLO      70$
1658
1659      ;VERIFY THAT DATA AREA COMES ON WITHIN 9 PROM STROBES
1660 017270 012702 000011      MOV      #9., R2      ;R2=9 STROBES
1661
1662      ;CLOCK PROM STROBE ON
1663 017274 130$:
1664 017274 004737 017700      JSR      PC, 300$
1665
1666      ;PROM STROBE ON CKECK IF DATA AREA ON
1667 017300 140$:
1668 017300 016004 000024      MOV      RMMR1(R0), R4      ;STORE RMMR1 AT R4
1669 017304 042704 177377      BIC      #^CPDA, R4
1670 017310 001020      BNE      160$
1671
1672      ;CLOCK PROM STROBE OFF
1673 017312 150$:
1674 017312 004737 020006      JSR      PC, 350$
1675 017316 005302      DEC      R2
1676 017320 001365      BNE      130$
1677
1678      ;ERROR-DATA AREA DIDN'T COME ON

```

```

1679 017322 012737 000400 001140      MOV    #PDA,$GDDAT      ;SETUP ERROR MESSAGE
1680 017330 005037 001142      CLR    $BDDAT
1681 017334 010037 001136      MOV    R0,$BDADR
1682 017340 062737 000024 001136      ADD    #RMMR1,$BDADR
1683 017346 104055      EMT    55
1684 017350 000541      BR     220$
1685
1686      ;VERIFY THAT DATA AREA IS ON FOR 256 CYCLES
1687 017352 160$:      MOV    #256.,R2
1688 017352 012702 000400
1689
1690      ;VERIFY THAT DATA AREA IS ON AND ECC IS OFF
1691 017356 170$:
1692 017356 016003 000024      MOV    RMMR1(R0),R3    ;STORE RMMR1 AT R3
1693 017362 042703 177357      BIC    #^C<PDA!ECC>,R3
1694 017366 022703 000400      CMP    #PDA,R3
1695 017372 001414      BEQ    180$            ;DATA AREA IS ON
1696 017374 010337 001142      MOV    R3,$BDDAT
1697 017400 012737 000400 001140      MOV    #PDA,$GDDAT
1698 017406 010037 001136      MOV    R0,$BDADR
1699 017412 062737 000024 001136      ADD    #RMMR1,$BDADR
1700 017420 104055      EMT    55
1701 017422 000514      BR     220$
1702
1703      ;CLOCK PROM STROBE OFF
1704 017424 180$:      JSR    PC,350$
1705 017424 004737 020006
1706
1707      ;CLOCK PROM STROBE ON
1708 017430 004737 017700      JSR    PC,300$
1709
1710      ;CONTINUE TIL COUNT ZERO
1711 017434 005302      DEC    R2
1712 017436 001347      BNE    170$
1713
1714      ;ECC SHOULD BE ENABLED WITHIN 4 CLOCK BITS
1715 017440 005002      CLR    R2
1716 017442 182$:
1717 017442 016003 000024      MOV    RMMR1(R0),R3    ;STORE RMMR1 AT R3
1718 017446 042703 177357      BIC    #^C<PDA!ECC>,R3
1719 017452 022703 000020      CMP    #ECC,R3
1720 017460 005202      BEQ    190$
1721 017462 012760 055401 000024      INC    R2
1722 017470 012760 051401 000024      MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1723 017476 022702 000007      MOV    #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1724 017502 101357      CMP    #7,R2
1725 017504 010337 001142      BHI    182$
1726 017510 012737 000020 001140      MOV    R3,$BDDAT
1727 017516 010037 001136      MOV    #ECC,$GDDAT
1728 017522 062737 000024 001136      MOV    R0,$BDADR
1729 017530 104056      ADD    #RMMR1,$BDADR
1730 017532 000450      EMT    56
1731      BR     220$
1732
1733      ;SAMPLE ECC CHARACTER FOR 32 BITS
1734 017534 190$:      MOV    #STMP0,R2

```

```

1735 017540 005003          195$: CLR      R3
1736 017542 012704 000020    MOV      #16.,R4
1737 017546          200$:          MOV      RMMR1(R0),R5 ;STORE RMMR1 AT R5
      017546 016005 000024    ASL      R3
1738 017552 006303          BIT      #MWD,R5
1739 017554 032705 000010    BNE     205$ ;BRANCH IF ECC BIT OFF (MWD INVERTED)
1740 017560 001002          BIS     #BIT0,R3
1741 017562 052703 000001    205$:          MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1742 017566          MOV      #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1743 017566 012760 055401 000024    DEC     R4
1744 017574 012760 051401 000024    BNE     200$
1745 017602 005304          MOV     R3,(R2)+
1746 017604 001360          CMP     #TMP0+4,R2
1747 017606 010322          BHI     195$
1748 017610 022702 001200
1749 017614 101351
1750
1751          ;VERIFY ECC CHARACTER GENERATED
1752 017616 022737 131177 001174    CMP     #131177,$TMP0
1753 017624 001004          BNE     210$
1754 017626 022737 175154 001176    CMP     #175154,$TMP1
1755 017634 001407          BEQ     220$
1756 017636 012737 131177 001200 210$: MOV     #131177,$TMP2
1757 017644 012737 175154 001202    MOV     #175154,$TMP3
1758 017652 104061          EMT     61
1759
1760 017654 000137 020112    220$: JMP     400$ ;GREAT ESCAPE
1761
1762 017660 000200          230$: .WORD   PHA
1763 017662 000002          .WORD   BB01
1764 017664 000200          .WORD   PHA
1765 017666 000002          .WORD   BB01
1766 017670 000000          .WORD   0
1767 017672 000002          .WORD   2
1768 017674 000000          .WORD   0
1769 017676 000000          .WORD   0
1770
1771 017700 012737 000021 020002 300$: MOV     #17.,320$ ;CLOCK PROM STROBE ON
1772 017706          305$:
      017706 012760 055401 000024    MOV     #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1773 017714 012760 051401 000024    MOV     #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1774 017722 016037 000024 020004    MOV     RMMR1(R0),330$ ;STORE RMMR1 AT 330$
1775 017730 042737 177737 020004    BIC     #^CWC,330$
1776 017736 001020          BNE     310$ ;EXIT IF PROM STROBE ON
1777 017740 005337 020002          DEC     320$
1778 017744 001360          BNE     305$
1779 017746 013737 020002 001142    MOV     320$,SBDAT ;SETUP ERROR MESSAGE
1780 017754 012737 000040 001140    MOV     #WC,$GDDAT
1781 017762 010037 001136          MOV     R0,$BDADR
1782 017766 062737 000024 001136    ADD     #RMMR1,$BDADR
1783 017774 104030          EMT     30
1784 017776 000445          BR      400$
1785 020000 000207          310$: RTS     PC
1786 020002 000000          320$: .WORD   0 ;MAX BIT COUNT
1787 020004 000000          330$: .WORD   0 ;RMMR1 CONTENTS
1788
1789 020006 012737 000021 020106 350$: MOV     #17.,370$
  
```

```

1790                                     ;CLOCK PROM STROBE OFF
1791 020014 355$: MOV #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
      020014 012760 055401 000024 MOV #MR1AAA,RMMR1(R0) ;LOAD RMMR1
1792 020022 012760 051401 000024 MOV RMMR1(R0),380$ ;STORE RMMR1 AT 380$
1793 020030 016037 000024 020110 BIC #^CWC,380$
1794 020036 042737 177737 020110 BEQ 360$
1795 020044 001417 DEC 370$
1796 020046 005337 020106 BNE 355$
1797 020052 001360 MOV 380$, $BDDAT
1798 020054 013737 020110 001142 CLR $GDDAT
1799 020062 005037 001140 MOV RO, $BDADR
1800 020066 010037 001136 ADD #RMMR1, $BDADR
1801 020072 062737 000024 001136 EMT 62
1802 020100 104062 BR 400$
1803 020102 000403 360$: RTS PC
1804 020104 000207
1805
1806 020106 000000 370$: .WORD 0 ;MAX BIT COUNT
1807 020110 000000 380$: .WORD 0 ;CONTENTS OF RMMR1
1808
1809 020112 400$: NOP ;SUB-TEST EXIT POINT
1810 020112 000240
1811
1812
::*****
:*TEST 24 ECC DETECTION TEST
::*****
TST24:
      020114 SCOPE ;SCOPE CALL
      020114 000004 NOP
      020116 000240 MOV #STACK,S? ;LOAD THE STACK POINTER
      020120 012706 001100 MOV $BASE,RO ;RO = UNIBUS ADDRESS
      020124 013700 001276 MOV TSTQUE,R1 ;R1 = POINTER TO DEVICE
      020130 013701 001462 MOV #24,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
      020134 012737 000024 001226
1813
1814 ;SETUP REGISTER OUTPUT BUFFER FOR READ DATA COMMAND
1815 020142 012737 001466 001442 MOV #822.,RMDCO ;LAST CYLINDER
1816 020150 013737 001330 001414 MOV LSTRK,RMDAO ;SET LAST TRACK AND
1817 020156 112737 000036 001414 MOVB #30.,RMDAO ;LAST SECTOR
1818 020164 012737 012000 001440 MOV #FMT16!HCI,RMOFO ;INHIBIT HEADER COMPARE
1819 020172 012702 054244 MOV #BUFONE,R2
1820 020176 010237 001412 MOV R2,RMBAO
1821 020202 012737 177400 001410 MOV #-256.,RMWCO ;256 WORDS (2'S COMP)
1822 020210 012737 000071 001406 MOV #RD!GO,RMCSTO ;READ DATA COMMAND
1823
1824 020216 20$: JSR PC,ENBSCH ;EXECUTE DATA COMMAND TO POINT WHERE
      020216 004737 023430 ;SEARCH IS ENABLED USING SUBROUTINE.
      BR 30$ ;BRANCH TO 30$ IF NO ERROR
      EMT
      JMP 190$
1825 020226 000137 021100
1826
1827 020232 30$: JSR PC,SCTCMP ;FORCE SECTOR COMPARE USING SUBROUTINE
      020232 004737 024302 BR 40$ ;BRANCH TO 40$ IF NO ERROR
      020236 000403 EMT
      020240 104000 JMP 190$
1828 020242 000137 021100

```



```

1829
1830 020246          40$:
      020246 004737 024410      JSR   PC,SETLFS      ;SET 'LOOKING FOR SYNC' USING SUBROUTINE
      020252 000403          BR    50$           ;BRANCH TO 50$ IF NO ERROR
      020254 104000          EMT
1831 020256 000137 021100      JMP   190$
1832
1833 020262          50$:
      020262 004737 024572      JSR   PC,CLKSNC      ;CLOCK THE SYNC PATTERN USING SUBROUTINE
      020266 000403          BR    55$           ;BR TO 55$ IF NO ERROR
      020270 104000          EMT
1834 020272 000137 021100      JMP   190$
1835
1836      ;HEADER COMPARE IS INHIBITED FOR THIS TEST
1837      ;'LOOKING FOR SYNC' SHOULD GO OFF WITHIN ONE CYCLE
1838 020276 012702 000002      55$:  MOV   #2,R2      ;ALLOW 2 PASSES THRU LOOP
1839 020302 000405          BR    70$
1840 020304 004737 021212      60$:  JSR   PC,350$      ;RESET PROM STROBE
1841 020310 004737 021104      JSR   PC,300$      ;SET PROM STROBE
1842 020314 000240          NOP
1843 020316          70$:
      020316 016003 000024      MOV   RMMR1(R0),R3  ;STORE RMMR1 AT R3
1844 020322 032703 002000      BIT   #PLFS,R3
1845 020326 001421          BEQ   80$           ;BRANCH IF LOOKING FOR SYNC OFF
1846 020330 005302          DEC   R2
1847 020332 001364          BNE   60$
1848
1849      ;ERROR-LOOKING FOR SYNC DID NOT RESET DURING HEADER
1850 020334 010337 001142      MOV   R3,$BDDAT    ;SETUP ERROR MESSAGE
1851 020340 042737 175777 001142  BIC   #^CPLFS,$BDDAT
1852 020346 005037 001140          CLR   $GDDAT
1853 020352 010037 001136      MOV   R0,$BDADR
1854 020356 062737 000024 001136  ADD   #RMMR1,$BDADR
1855 020364 104040          EMT   40
1856 020366 000137 021100      JMP   190$
1857
1858      ;LOOKING FOR SYNC SHOULD COME ON FOR DATA AREA WITHIN 9 PROM CYCLES
1859 020372          80$:
1860 020372 012702 000011      MOV   #9,R2
1861 020376 004737 021212      85$:  JSR   PC,350$      ;SET STROBE OFF
1862 020402 004737 021104      JSR   PC,300$      ;SET PROM STROBE
1863 020406 016003 000024      MOV   RMMR1(R0),R3 ;STORE RMMR1 AT R3
1864 020412 032703 002000      BIT   #PLFS,R3
1865 020416 001022          BNE   90$           ;BRANCH IF SYNC ENABLED
1866 020420 005302          DEC   R2
1867 020422 001365          BNE   85$
1868
1869      ;ERROR-CAN'T LOOKING FOR SYNC DURING DATA AREA
1870 020424 012737 002000 001140  MOV   #PLFS,$GDDAT
1871 020432 010337 001142      MOV   R3,$BDDAT
1872
1873 020436 042737 175777 001142  BIC   #^CPLFS,$BDDAT
1874 020444 010037 001136      MOV   R0,$BDADR
1875 020450 062737 000024 001136  ADD   #RMMR1,$BDADR
1876 020456 104035          EMT   35
1877 020460 000137 021100      JMP   190$
1878 020464          90$:

```

```

1879 020464 004737 021212      JSR    PC,350$      ;RESET PROM STROBE
1880 020470 004737 024572      JSR    PC,CLKSNC   ;CLOCK THE SYNC PATTERN USING SUBROUTINE
      020474 000402      BR     100$        ;BR TO 100$ IF NO ERROR
      020476 104000      EMT
1881 020500 000577      BR     190$
1882
1883      ;SIMULATE READ DATA
1884 020502 100$:
1885 020502 012702 000400      MOV    #256.,R2    ;WORD COUNT
1886 020506 012703 000020      MOV    #16.,R3     ;BIT COUNT
1887 020512 115$:
      020512 012760 057401 000024      MOV    #MMR1AAA!MRD!MCLK,RMMR1(R0) ;LOAD RMMR1
1888 020520 012760 053401 000024      MOV    #MMR1AAA!MRD,RMMR1(R0) ;LOAD RMMR1
1889 020526 005303      DEC    R3
1890 020530 001370      BNE   115$
1891 020532 005302      DEC    R2
1892 020534 001364      BNE   110$
1893
1894      ;SIMULATE ECC PATTERN
1895 020536 012737 177446 001174      MOV    #177446,$TMP0 ;FIRST ECC WORD
1896 020544 012737 015457 001176      MOV    #015457,$TMP1 ;SECOND ECC WORD
1897 020552 012702 001174      MOV    #TMP0,R2
1898 020556 012703 000020      120$: MOV    #16.,R3
1899 020562 012704 051401      125$: MOV    #MMR1AAA,R4
1900 020566 000241      CLC
1901 020570 006012      ROR   (R2)
1902 020572 103002      BCC   130$
1903 020574 052704 002000      BIS   #MRD,R4
1904 020600 130$:
      020600 010460 000024      MOV    R4,RMMR1(R0) ;LOAD RMMR1
1905 020604 052704 004000      BIS   #MCLK,R4
1906 020610 010460 000024      MOV    R4,RMMR1(R0) ;LOAD RMMR1
1907 020614 042704 004000      BIC   #MCLK,R4
1908 020620 010460 000024      MOV    R4,RMMR1(R0) ;LOAD RMMR1
1909 020624 005303      DEC    R3
1910 020626 001355      BNE   125$ ;CLOCK A WORD
1911 020630 062702 000002      ADD   #2,R2
1912 020634 022702 001176      CMP   #TMP1,R2
1913 020640 001746      BEQ   120$ ;CLOCK THE NEXT WORD
1914
1915      ;VERIFY DATA AREA AND READ GATE RESET
1916 020642 012702 000005      MOV   #5,R2
1917
1918 020646 004737 021104      140$: JSR   PC,300$ ;SET PROM STROBE
1919 020652 004737 021212      JSR   PC,350$ ;CLEAR PROM STROBE
1920 020656 016003 000024      MOV   RMMR1(R0),R3 ;STORE RMMR1 AT R3
1921 020662 032703 000400      BIT   #PDA,R3
1922 020666 001417      BEQ   150$
1923 020670 005302      DEC   R2
1924 020672 001365      BNE   140$
1925 020674 042703 177377      BIC   #^CPDA,R3 ;SETUP ERROR MESSAGE
1926 020700 010337 001142      MOV   R3,$BDDAT
1927 020704 005037 001140      CLR   $GDDAT
1928 020710 010037 001136      MOV   R0,$BDADR
1929 020714 062737 000024 001136      ADD   #RMMR1,$BDADR
1930 020722 104063      EMT   63
1931 020724 000465      BR    190$

```

```

1932 020726          150$:
1933 020726 016003 000040      MOV    RMMR2(R0),R3      ;STORE RMMR2 AT R3
1934 020732 042703 176000      BIC    #^C1777,R3
1935 020736 001413              BEQ    160$
1936 020740 010337 001142      MOV    R3,$BDDAT
1937 020744 005037 001140      CLR    $GDDAT
1938 020750 010037 001136      MOV    R0,$BDADR
1939 020754 062737 000040 001136  ADD    #RMMR2,$BDADR
1940 020762 104053              EMT    53
1941 020764 000445              BR     190$
1942
1943
1944 020766          ;VERIFY THERE ARE NO ECC ERRORS
1945 020766 016003 000014      160$:  MOV    RMER1(R0),R3      ;STORE RMER1 AT R3
1946 020772 042703 077677      BIC    #^C<DCK!ECH>,R3
1947 020776 012737 000000 001140  MOV    #000000,$GDDAT
1948 021004 023703 001140      CMP    $GDDAT,R3
1949 021010 001411              BEQ    170$
1950 021012 010337 001142      MOV    R3,$BDDAT
1951 021016 010037 001136      MOV    R0,$BDADR
1952 021022 062737 000014 001136  ADD    #RMER1,$BDADR
1953 021030 104064              EMT    64
1954 021032 000422              BR     190$
1955
1956          ;VERIFY DATA BUFFER
1957 021034          170$:
1958 021034 012702 054244      MOV    #BUFONE,R2      ;DATA WAS STORED HERE
1959 021040 012703 177777      MOV    #177777,R3      ;EACH WORD ALL ONES
1960 021044 012704 000400      MOV    #256.,R4
1961 021050 022203          180$:  CMP    (R2)+,R3
1962 021052 001003          BNE    185$
1963 021054 005304          DEC    R4
1964 021056 001374          BNE    180$
1965 021060 000407          BR     190$
1966 021062 014237 001142      185$:  MOV    -(R2),$BDDAT      ;EXIT IF ALL THE LOCATIONS CHECKED
1967 021066 010337 001140      MOV    R3,$GDDAT      ;SETUP ERROR MESSAGE
1968 021072 010237 001136      MOV    R2,$BDADR
1969 021076 104065          EMT    65
1970 021100 000137 021316      190$:  JMP    400$            ;GREAT ESCAPE
1971 021104 012737 000021 021206  300$:  MOV    #17.,320$
1972          ;CLOCK PROM STROBE ON
1973 021112          305$:
1974 021112 012760 055401 000024      MOV    #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1975 021120 012760 051401 000024      MOV    #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
1976 021126 016037 000024 021210  MOV    RMMR1(R0),330$ ;STORE RMMR1 AT 330$
1977 021134 042737 177737 021210  BIC    #^CWC,330$
1978 021142 001020          BNE    310$            ;EXIT IF PROM STROBE ON
1979 021144 005337 021206          DEC    320$
1980 021150 001360          BNE    305$
1981 021152 013737 021206 001142      MOV    320,$BDDAT      ;SETUP ERROR MESSAGE
1982 021160 012737 000040 001140      MOV    #WC,$GDDAT
1983 021166 010037 001136          MOV    R0,$BDADR
1984 021172 062737 000024 001136  ADD    #RMMR1,$BDADR
1985 021200 104030          EMT    30
1986 021202 000445          BR     400$
1987 021204 000207          310$:  RTS    PC
1987 021206 000000          320$:  .WORD 0

```

```

1988
1989 021210 000000          330$: .WORD 0
1990
1991 021212 012737 000021 021312 350$: MOV #17.,370$ ;CLOCK THE STROBE OFF
1992 021220          355$:
    021220 012760 055401 000024      MOV #R1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
1993 021226 012760 051401 000024      MOV #R1AAA,RMMR1(R0) ;LOAD RMMR1
1994 021234 016037 000024 021314      MOV RMMR1(R0),380$ ;STORE RMMR1 AT 380$
1995 021242 042737 177737 021314      BIC #^CWC,380$
1996 021250 001417          BEQ 360$
1997 021252 005337 021312          DEC 370$
1998 021256 001360          BNE 355$
1999 021260 013737 021314 001142      MOV 380$,SBDDAT
2000 021266 005037 001140          CLR $GDDAT
2001 021272 010037 001136          MOV R0,$BDADR
2002 021276 062737 000024 001136      ADD #RMMR1,$BDADR
2003 021304 104062          EMT 62
2004 021306 000403          BR 400$
2005 021310 000207          360$: RTS PC
2006 021312 000000          370$: .WORD 0
2007 021314 000000          380$: .WORD 0
2008 021316 000240          400$: NOP ;SUB-TEST EXIT POINT
2009
  
```

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 'READY' 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    READY         ;YES, JUMP TO READY
        MOV    #TSTQUE+2,TSTQUE ;INITIALIZE POINTER TO FIRST DEVICE IN
                                ;TEST QUE TABLE
```

```
.S9TTL 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 READY
```

```
021356      $EOP:
021356      NOP
021360      CLR    $TSTNM      ;;ZERO THE TEST NUMBER
021364      CLR    $TIMES     ;;ZERO THE NUMBER OF ITERATIONS
021370      INC    $PASS      ;;INCREMENT THE PASS NUMBER
021374      BIC    #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
021402      DEC    (PC)+      ;;LOOP?
021404      $EOPCT: .WORD 1
021406      BGT    $DOAGN     ;;YES
021410      MOV    (PC)+,@(PC)+ ;;RESTORE COUNTER
021412      $ENDCT: .WORD 1
021414      $EOPCT
021416      TYPE   ,65$      ;;TYPE ASCIZ STRING
021422      BR    64$        ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <12><15>/END PASS #/
64$:
021442      MOV    $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
021442      ;;TYPE PASS NUMBER
021446      TYPDS  ;;GO TYPE--DECIMAL ASCII WITH SIGN
021450      TYPE   ,67$      ;;TYPE ASCIZ STRING
021454      BR    66$        ;;GET OVER THE ASCIZ
;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
66$:
021520      MOV    $ERTTL,-(SP) ;;SAVE $ERTTL FOR TYPEOUT
021520      ;;TOTAL NUMBER OF ERRORS
021524      TYPDS  ;;GO TYPE--DECIMAL ASCII WITH SIGN
021526      TYPE   ,SCLRF    ;;TYPE CARRIAGE RETURN, LINE FEED
021532      CLR    $ERTTL    ;;CLEAR ERROR TOTAL
021536      $GET42: MOV    @#42,RO ;;GET MONITOR ADDRESS
```

END OF PASS ROUTINE

```
021542 001405          BEQ      $DOAGN      ;;BRANCH IF NO MONITOR
021544 000005          RFSSET          ;;CLEAR THE WORLD
021546 004710  SENDAD: JSR      PC,(R0)    ;;GO TO MONITOR
021550 000240          NOP              ;;SAVE ROOM
021552 000240          NOP              ;;FOR
021554 000240          NOP              ;;ACT11
021556 000137          $DOAGN:          ;;
021556 005242          JMP      @(PC)+    ;;RETURN
021560 005242          $RTNAD: .WORD    READY
021562 377 000 $ENULL: .BYTE    -1,-1,0  ;;NULL CHARACTER STRING
          .EVEN
```

21

```
1 .SBTTL SUBROUTINES
2 :*****
3 .SBTTL ERROR TYPEOUT ROUTINE
4
5 :*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
6 :*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
7 :*
8 :* .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
9 :*PRINTED ON THE FIRST LINE;
10 :* .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
11 :*ONE OR MORE SUCCEEDING LINES;
12 :* .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
13 :*ARE PRINTED AFTER THE ERROR MESSAGE.
14
15 021566
16 021566 104414
17 021570 032777 020000 157356
18 021576 001402
19 021600 000137 022402
20
21 :TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
22 021604 104401 001217 1$: TYPE ,SCLF
23 021610 104401 022416 TYPE ,ERTY00 ;TYPE 'UNIT#'
24 021614 013746 001234 MOV $UNIT,-(SP) ;;SAVE $UNIT FOR TYPEOUT
;TYPE UNIT NUMBER
;GO TYPE--OCTAL ASCII
;TYPE 3 DIGIT(S)
;SUPPRESS LEADING ZEROS
021620 104403 TYPOS
021622 003 .BYTE 3
021623 000 .BYTE 0
25
26 :TYPE 'DRIVE TYPE' RM05, RM03 OR RM02 FOR UNIT UNDER TEST
27 021624 016000 000026 MOV RMDT(R0),R0 ;STORE RMDT AT R0
28 021630 042700 177740 BIC #177740,R0 ;SAVE DRIVE TYPE BITS AND
29 021634 012737 022465 021706 MOV #SRM03,3$ ;GET ASCII DRIVE TYPE
30 021642 022700 000024 CMP #24,R0 ;IS DEVICE AN RM03 ?
31 021646 001414 BEQ 2$ ;YES !!
32
33 021650 012737 022460 021706 MOV #SRM02,3$ ;SAVE ASCII DRIVE TYPE
34 021656 022700 000025 CMP #25,R0 ;IS DEVICE AN RM02 ?
35 021662 001406 BEQ 2$ ;YES !!
36
37 021664 012737 022472 021706 MOV #SRM05,3$ ;SAVE ASCII DRIVE TYPE
38 021672 022700 000027 CMP #27,R0 ;IS DEVICE AN RM05 ?
39 021676 001004 BNE 4$ ;NO !!
40 021700 104401 022454 2$: TYPE ,ERTY05 ;TYPE '' - ''
41 021704 104401 TYPE ;TYPE DRIVE TYPE
42 021706 000000 3$: .WORD 0 ;DRIVE TYPE MESSAGE IS STORED HERE
43
44 :TYPE TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER
45 021710 005037 022406 4$: CLR TSTNMB ;LOAD TEST NUMBER FOR
46 021714 013737 001226 022406 MOV $TESTN,TSTNMB
47 021722 104401 022424 TYPE ,ERTY01 ;TYPE 'TST#'
48 021726 013746 022406 MOV TSTNMB,-(SP) ;;SAVE TSTNMB FOR TYPEOUT
;TYPE TEST NUMBER
;GO TYPE--OCTAL ASCII
;TYPE 3 DIGIT(S)
;SUPPRESS LEADING ZEROS
021732 104403 TYPOS
021734 003 .BYTE 3
021735 000 .BYTE 0
49 021736 005037 022410 CLR ERRNMB ;LOAD ERROR NUMBER FOR
```

```

50 021742 113737 001130 022410      MOVB    $ITEMB,ERRNMB    ;TYPEOUT
51 021750 001406                    BEQ     5$              ;SKIP IF NO ERROR CALLED
52 021752 104401 022434              TYPE    ,ERTY02         ;TYPE 'ERR#'
53 021756 013746 022410              MOV     ERRNMB,-(SP)    ;SAVE ERRNMB FOR TYPEOUT
                                ;TYPE ERROR NUMBER
                                ;GO TYPE--OCTAL ASCII
                                ;TYPE 3 DIGIT(S)
                                ;SUPPRESS LEADING ZEROS
                                ;TYPE 'PC='
021762 104403                    TYPOS
021764 003                          .BYTE  3
021765 000                          .BYTE  0
54 021766 104401 022443              5$:    TYPE    ,ERTY03
55 021772 013746 001132              MOV     $ERRPC,-(SP)   ;SAVE $ERRPC FOR TYPEOUT
                                ;TYPE PROGRAM COUNTER
                                ;GO TYPE--OCTAL ASCII
                                ;TYPE 6 DIGIT(S)
                                ;TYPE LEADING ZEROS
021776 104403                    TYPOS
022000 006                          .BYTE  6
022001 001                          .BYTE  1

56
57
58 022002 005737 022410              6$:    ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
59 022006 001575                    TST    ERRNMB          ;WAS AN ERROR CALLED?
60 022010 104401 001217              BEQ    25$            ;NO!!
61 022014 105037 022414              TYPE    , $CRLF        ;YES-TYPE CRLF
62 022020 105037 022415              CLRB   BOTFLG        ;CLEAR BOT FLAG
63 022024 013700 022410              CLRB   CHRCNT        ;CLEAR CHARACTER COUNTER
64 022030 006300                    MOV     ERRNMB,R0    ;R0 POINTS TO FIRST OF
65 022032 006300                    ASL    R0             ;FOUR ENTRIES IN ERROR
66 022034 006300                    ASL    R0             ;TABLE
67 022036 062700 001526              ADD    # $ERRTB-8.,R0
68 022042 011001                    MOV     (R0),R1      ;R1 POINTS TO ERROR MESSAGE
69
70 022044 001507                    BEQ    16$            ;BRANCH IF NO ERROR MESSAGE
71
72
73 022046 012102                    7$:    MOV     (R1)+,R2    ;R2=ADDRESS OF MESSAGE STRING
74 022050 001505                    BEQ    16$            ;BRANCH IF END OF MESSAGE
75 022052 010237 022220              MOV     R2,15$        ;LOAD ADDRESS OF STRING
76 022056 005037 022412              CLR    BOTADR        ;CLEAR BOT ADDRESS
77 022062 112203                    8$:    MOVB   (R2)+,R3    ;END OF STRING??
78 022064 001454                    BEQ    14$            ;YES!!
79 022066 122703 000015              CMPB   #CR,R3        ;CARRIAGE RETURN??
80 022072 001003                    BNE    9$            ;NO!!
81 022074 105037 022415              CLRB   CHRCNT        ;YES-CLEAR CHAR COUNT
82 022100 000770                    BR     8$            ;GET NEXT CHARACTER
83 022102 122703 000012              9$:    CMPB   #LF,R3    ;LINE FEED??
84 022106 001765                    BEQ    8$            ;YES-GET NEXT CHARACTER
85 022110 122703 000011              CMPB   #HT,R3        ;HORIZONTAL TAB??
86 022114 001007                    BNE    11$           ;NO!!
87 022116 105237 022415              10$:   INCB   CHRCNT      ;ADJUST CHARACTER COUNT
88 022122 132737 000007 022415      BITB   #7,CHRCNT
89 022130 001372                    BNE    10$
90 022132 000407                    BR     12$
91 022134 105237 022415              11$:   INCB   CHRCNT      ;INCREMENT CHARACTER COUNT
92 022140 122703 000040              CMPB   #' ,R3        ;SPACE??
93 022144 001002                    BNE    12$           ;NO!!
94 022146 010237 022412              MOV     R2,BOTADR    ;SAVE ADDRESS OF SPACE
95 022152 122737 000100 022415      12$:   CMPB   #64.,CHRCNT ;END OF LINE??
96 022160 103340                    BHS    8$            ;NO!!
97 022162 013704 022412              MOV     BOTADR,R4    ;GET ADDRESS OF LAST SPACE
98 022166 001007                    BNE    13$           ;BRANCH IF SPACE DETECTED

```



```

99 022170 104401 001217      TYPE      , $CRLF      ;TYPE CRLF
100 022174 105037 022415      CLRB      CHRCNT      ;CLEAR CHARACTER COUNT
101 022200 013702 022220      MOV       15$,R2     ;SET UP R2 FOR TESTING
102 022204 000726                BR          8$
103 022206 105044                CLRB      -(R4)      ;REPLACE SPACE
104 022210 112737 177777 022414  MOVB      #-1,BOTFLG ;SET BOT FLAG
105 022216 104401                TYPE      ;TYPE ERROR MESSAGE STRING
106 022220 000000                .WORD     ;STRING ADDRESS GOES HERE
107 022222 105737 022414      TSTB     BOTFLG     ;WAS STRING TRUNCATED??
108 022226 001707                BEQ       7$        ;NO!!
109 022230 104401 001217      TYPE      , $CRLF      ;YES-TYPE CRLF
110 022234 105037 022414      CLRB     BOTFLG     ;CLEAR BOT FLAG
111 022240 105037 022415      CLRB     CHRCNT     ;CLEAR CHARACTER COUNT
112 022244 013702 022412      MOV      BOTADR,R2  ;SETUP R2 FOR TESTING
113 022250 010237 022220      MOV      R2,15$     ;SETUP 15$ FOR TYPING
114 022254 112742 000040      MOVB     #' ',-(R2) ;RESTORE SPACE
115 022260 105722                TSTB     (R2)+      ;RESTORE R2
116 022262 000677                BR          8$      ;TYPE REST OF STRING
117
118                                ;TYPE ERROR HEADER AND ERROR DATA
119 022264                16$:
120 022264 016001 000002      17$: MOV      2(R0),R1  ;R1 POINTS TO ERROR HEADER TABLE
121 022270 001444                BEQ      25$        ;BRANCH IF NO HEADER
122 022272 104401 001217      TYPE      , $CRLF      ;(ASSUME NO DATA)
123 022276 016002 000004      MOV      4(R0),R2   ;R2 POINTS TO DATA ADDRESS TABLE
124 022302 016003 000006      MOV      6(R0),R3   ;R3 POINTS TO FORMAT TABLE
125 022306 012137 022316      18$: MOV      (R1)+,19$ ;PUT HEADER ADDRESS FOR TYPE
126 022312 001433                BEQ      25$        ;BRANCH IF END OF HEADERS
127                                ;(ASSUME END OF DATA)
128 022314 104401                TYPE
129 022316 000000                .WORD     0          ;HEADER ADDRESS GOES HERE
130 022320 104401 001217      TYPE      , $CRLF
131 022324 005702                TST      R2         ;DATA WITH HEADER??
132 022326 001767                BEQ      18$        ;NO!!
133 022330 012204      MOV      (R2)+,R4   ;R4 POINTS TO DATA ADDRESS
134 022332 012305      MOV      (R3)+,R5   ;R5 POINTS TO FORMAT
135 022334 105725      20$: TSTB     (R5)+      ;WHAT KIND OF DATA??
136 022336 100407                BMI      22$        ;BINARY
137 022340 001403                BEQ      21$        ;OCTAL
138 022342 013446      MOV      @ (R4)+,-(SP) ;DECIMAL
139 022344 104405                TYPDS
140 022346 000405                BR          23$
141 022350 013446      21$: MOV      @ (R4)+,-(SP)
142 022352 104402                TYPOC
143 022354 000402                BR          23$
144 022356 013446      22$: MOV      @ (R4)+,-(SP)
145 022360 104406                TYPBN
146 022362 005714      23$: TST      (R4)      ;MORE DATA??
147 022364 001403                BEQ      24$        ;NO!!
148 022366 104401 022451      TYPE      ,ERTY04    ;YES-TYPE 2 SPACES
149 022372 000760                BR          20$      ;AND CONTINUE
150 022374 104401 001217      24$: TYPE      , $CRLF ;TYPE ONE BLANK LINE
151 022400 000742                BR          18$      ;BEFORE NEXT HEADER
152 022402 104415      25$: RESREG
153 022404 000207                RTS      PC
154
155 022406 000000      TSTNMB: .WORD     0 ;TEST NUMBER
    
```



```

1          .SBTTL  CLOCK SUBROUTINES
2
3          ;ROUTINE TO SIZE FOR CLOCKS (KW11-L OR KW11-P)
4
5 022500 000240          SIZCLK: NOP
6 022502 013746 000004  MOV      ERRVEC,-(SP)      ;;PUSH ERRVEC ON STACK
7 022506 013746 000006  MOV      ERRVEC+2,-(SP)    ;;PUSH ERRVEC+2 ON STACK
8 022512 012737 022576 000004  MOV      #10$,ERRVEC      ;LOAD 04 TRAP VECTORS
9 022520 012737 000300 000006  MOV      #PR6,ERRVEC+2
10
11          ;SEE IF A KW11-P CLOCK IS PRESENT - GO TO 10$ IF NOT PRESENT
12 022526 005777 156754  TST      @SLPCSR          ;TEST FOR P CLOCK
13 022532 012737 022740 001532  MOV      #PCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
14 022540 012737 023062 001534  MOV      #PSTOP,STOP     ;LOAD STOP ADDRESS
15 022546 012777 023026 156736  MOV      #PCOUNT,@SLPVEC ;LOAD P CLOCK INTERRUPT VECTOR
16 022554 012777 000300 156732  MOV      #PR6,@SLPVEC+2
17 022562 013777 001522 156730  MOV      $LLVEC+2,@$LLVEC;CLEAR L CLOCK INTERRUPT VECTOR
18 022570 005077 156726  CLR      @$LLVEC+2
19 022574 000454          BR      30$
20 022576 012716 022604 10$:  MOV      #15$, (SP)      ;DUMMY RTI ADDRESS
21 022602 000002          RTI      ;RESTORE PRIORITY
22
23          ;NO P CLOCK-SEE IF L CLOCK IS PRESENT-GO TO 20$ IF NOT PRESET
24 022604          15$:
25 022604 012737 022662 000004  MOV      #20$,ERRVEC     ;CHANGE 04 TRAP VECTOR
26 022612 005777 156700  TST      @$LLCSR          ;TEST FOR L CLOCK
27 022616 012737 022756 001532  MOV      #LCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
28 022624 012737 023070 001534  MOV      #LSTOP,STOP     ;LOAD STOP ADDRESS
29 022632 012777 023026 156660  MOV      #LCOUNT,@$LLVEC ;LOAD L CLOCK INTERRUPT VECTOR
30 022640 012777 000300 156654  MOV      #PR6,@$LLVEC+2
31 022646 013777 001514 156636  MOV      $LPVEC+2,@$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
32 022654 005077 156634  CLR      @$LPVEC+2
33 022660 000422          BR      30$
34 022662 012716 022670 20$:  MOV      #25$, (SP)      ;DUMMY RTI ADDRESS
35 022666 000002          RTI      ;RESTORE PRIORITY
36
37          ;NO CLOCK AVAILABLE - AUGMENT RETURN ADDRESS
38 022670          25$:
39 022670 005037 001532  CLR      CLOCK           ;CLEAR SUBROUTINE ADDRESS
40 022674 012737 001514 001512  MOV      #$LPVEC+2,$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
41 022702 005037 001514  CLR      $LPVEC+2
42 022706 012737 001522 001520  MOV      #$LLVEC+2,$LLVEC;CLEAR L CLOCK INTERRUPT VECTOR
43 022714 005037 001522  CLR      $LLVEC+2
44 022720 062766 000002 000004  ADD      #2,4(SP)        ;CHANGE RETURN ADDRESS
45
46 022726          30$:
47 022726 012637 000006  MOV      (SP)+,ERRVEC+2  ;;POP STACK INTO ERRVEC+2
48 022732 012637 000004  MOV      (SP)+,ERRVEC   ;;POP STACK INTO ERRVEC
49 022736 000207  RTS      PC
50
51          ;ROUTINES TO START THE CLOCK (KW11-L OR KW11-P)
52 022740 012777 177777 156542  PCLOCK: MOV      #-1,@$SLPCSB ;LOAD COUNT SET BUFFER
53 022746 012777 000135 156532  MOV      #135,@$SLPCSR  ;LOAD CONTROL REGISTER
54 022754 000403          BR      PLCLK          ;GO TO COMMON CODE
55
56 022756 012777 000100 156532  LCLOCK: MOV      #100,@$LLCSR ;LOAD CONTROL REGISTER

```

```

57
58 022764 005037 001526      PLCLK: CLR      TIME      ;CLEAR TIMER COUNT
59 022770 104400              TRAP                      ;:PUSH OLD PSW AND PC ON STACK
    022772 012605              MOV      (SP)+,R5         ;:SAVE THE PSW IN R5
60 022774 010537 001524      MOV      R5,$PSW         ;:SAVE PRIORITY
61 023000 042705 177437      BIC      #^CPR7,R5       ;:MASK X
62 023004 022705 000300      CMP      #PR6,R5        ;:IS PRIORITY TOO HIGH??
63 023010 101005              BHI      40$             ;:NO!!
64 023012 012746 000240      MOV      #PR5,-(SP)      ;:PUT NEW PS ON STACK
    023016 012746 023024      MOV      #30$,-(SP)     ;:PUT NEW PC ON STACK
    023022 000002              RTI                      ;:POP NEW PC AND PS
    023024
65 023024 000207      30$:
    40$:      RTS      PC
66
67      ;ROUTINES TO HANDLE CLOCK INTERRUPTS (KW11-L OR KW11-P)
68
69 023026      PCOUNT:
70 023026      LCOUNT:
71 023026 062737 000021 001526      ADD      #17.,TIME      ;ADD 17MS TO ELAPSED TIME
72 023034 103003              BCC      10$            ;:BRANCH IF NO OVERFLOW
73 023036 012737 177777 001526      MOV      #-1.,TIME      ;:RESTORE MAXIMUM COUNT
74 023044 162737 000021 001530      10$:      SUB      #17.,WATCH   ;:DECREMENT REMAINING TIME
75 023052 100002              BPL      20$            ;:BRANCH IF POSITIVE
76 023054 005037 001530      CLR      WATCH          ;:CLEAR REMAINING TIME
77 023060 000002      20$:      RTI                      ;:RETURN TO USER
78
79      ;ROUTINES TO STOP THE CLOCK (KW11-L OR KW11-P)
80
81 023062 005077 156420      PSTOP: CLR      @SLPCSR   ;STOP P CLOCK
82 023066 000402              BR      PLSTP          ;GO TO COMMON STOP CODE
83
84 023070 005077 156422      LSTOP: CLR      @SLLCSR   ;STOP L CLOCK
85
86 023074      PLSTP:
87 023074 013746 001524      MOV      $PSW,-(SP)     ;:PUT NEW PS ON STACK
    023100 012746 023106      MOV      #10$,-(SP)    ;:PUT NEW PC ON STACK
    023104 000002              RTI                      ;:POP NEW PC AND PS
    023106
88 023106 000207      10$:
89              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

.SBTTL SET VOLUME VALID SUBROUTINE

;THIS SUBROUTINE INITIALIZES THE SUBSYSTEM AND SETS VOLUME VALID,
 ;RETURNING WITH THE DRIVE STILL IN DIAGNOSTIC MODE. THE SUBROUTINE
 ;RETURNS TO THE WORD FOLLOWING THE CALL, EXCEPT WHEN AN ERROR IS
 ;DETECTED, IN WHICH CASE IT RETURNS TO THE SECOND WORD FOLLOWING THE
 ;CALL.

```

:CALL: JSR    PC,SETVV      JUMP TO SUBROUTINE
:      BR     ??           RETURN HERE IF NO ERROR
:      ERROR          RETURN HERE IF ERROR
  
```

SETVV:

```

JSR    PC,CNTCLR          ;GO CLEAR CONTROLLER
MOV    #DMD,RMMR1(R0)    ;LOAD RMMR1
MOV    #DMD!MUR,RMMR1(R0) ;LOAD RMMR1
MOV    #0,RMER1(R0)      ;LOAD RMER1
MOV    #0,RMER2(R0)      ;LOAD RMER2
MOV    #PACACK!GO,RMCS1(R0) ;LOAD RMCS1
MOV    RMD5(R0),SBDDAT   ;STORE RMD5 AT SBDDAT
BIC    #^CVV,SBDDAT
BNE    10$              ;BRANCH IF VOLUME VALID SET
MOV    R0,$BDADR         ;SETUP FOR ERROR MSG
ADD    #RMD5,$BDADR
MOV    #VV,$GDDAT
ADD    #2,(SP)           ;MOVE RETURN ADDRESS TO ERROR
MOVB   #100,a(SP)       ;WRITE ERROR NUMBER
MOV    #PACACK,$TMPO
10$:   RTS    PC          ;RETURN
  
```

```

023110
023110 004737 023334
023114 012760 000001 000024
023122 012760 001001 000024
023130 012760 000000 000014
023136 012760 000000 000042
023144 012760 000023 000000
023152 016037 000012 001142
023160 042737 177677 001142
023166 001020
023170 010037 001136
023174 062737 000012 001136
023202 012737 000100 001140
023210 062716 000002
023214 112776 000100 000000
023222 012737 000022 001174
023230 000207
  
```

```

1      .SBTTL SET OFFSET MODE SUBROUTINE
2
3      ;THIS SUBROUTINE EXECUTES AN OFFSET COMMAND AND VERIFIES THAT OFFSET
4      ;MODE SETS. THE DRIVE SHOULD BE IN DIAGNOSTIC MODE WHEN CALLING THE
5      ;SUBROUTINE, WHICH WILL LEAVE DMD ON. THE SUBROUTINE RETURNS TO THE
6      ;WORD FOLLOWING THE CALL UNLESS THERE IS AN ERROR, IN WHICH CASE IT
7      ;RETURNS TO THE SECOND WORD FOLLOWING THE CALL
8
9      ;CALL: JSR      PC,SETOM      JUMP TO SUBROUTINE
10     ;        BR      ??          RETURN HERE IF NO ERROR
11     ;        ERROR   RETURN HERE IF ERROR
12
13     023232 SETOM:
14     023232 012760 001001 000024 MOV      #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
15     023240 012760 000000 000014 MOV      #0,RMER1(R0)          ;LOAD RMER1
16     023246 012760 000000 000042 MOV      #0,RMER2(R0)          ;LOAD RMER2
17     023254 012760 000015 000000 MOV      #OFFSET!GO, RMCS1(R0)   ;LOAD RMCS1
18     023262 016037 000012 001142 MOV      RMDS(R0), $BDDAT ;STORE RMDS AT $BDDAT
19     023270 042737 177776 001142 BIC      #^COM,$BDDAT
20     023276 001015 BNE      10$ ;BRANCH IF OFFSET ON
21     023300 012737 000001 001140 MOV      #OM,$GDDAT
22     023306 010037 001136 MOV      R0,$BDADR
23     023312 062737 000012 001136 ADD      #RMDS,$BDADR
24     023320 062716 000002 ADD      #2,(SP) ;MOVE RETURN ADDRESS TO ERROR
25     023324 112776 000111 000000 MOV      #111,@(SP) ;WRITE ERROR NUMBER
26     023332
27     023332 000207 10$: RTS      PC ;RETURN TO USER
28

```

```

1      .SBTTL CLEAR CONTROLLER SUBROUTINE
2
3      ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
4      ;AND DRIVES, THEN SELECTS THE DRIVE.
5      ;CALL:
6      ;      JSR      PC,CNTCLR      ;CALL TO ROUTINE
7
8      CNTCLR:
9      023334      010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
10     023336      013700      001276      MOV      $BASE,R0      ;;R0 = UNIBUS BASE ADDRESS
11     023342      012760      000040      000010      MOV      #CLR,RMCS2(R0) ;CLEAR MASSBUS
12     023350      117760      156106      000010      MOV      @TSTQUE,RMCS2(R0) ;SELECT DEVICE UNDER TEST
13     023356      012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
14     023360      000207      RTS      PC      ;RETURN
15
16     .SBTTL SHUTDOWN SUBROUTINE
17     SHUT:
18     023362      104401      023370      TYPE      ,65$      ;;TYPE ASCIZ STRING
19     023366      000411      BR      64$      ;;GET OVER THE ASCIZ
20     ;;65$: .ASCIZ <CRLF>@TEST WAS HALTED@<CRLF>
21     64$:
22     023412      005737      000042      TST      @#42      ;RUNNING CHAIN MODE OR ACT11 AUTO ACCEPT ?
23     023416      001402      BEQ     10$      ;NO !!
24     023420      000137      021356      JMP      $EOP      ;YES - GO TO END OF PASS
25     10$: JMP      START      ;GO TO START
  
```

```
1          .SBTTL  ENABLE SEARCH SUBROUTINE
2
3 023430   ENBSCH:
4
5          ;THE REGISTER OUTPUT BUFFER SHOULD CONTAIN:
6          :      RMDAO  = DESIRED SECTOR AND TRACK ADDRESS
7          :      RMDCO  = DESIRED CYLINDER ADDRESS
8          :      RMOFO  = FORMAT, ECI, HCI
9          :      RMBAO  = BUFFER ADDRESS
10         :      RMWCO  = WORD COUNT
11         :      RMCS10 = FUNCTION CODE
12
13         ;NOTE: OFFSET IS NOT ENABLED WHEN USING THIS SUBROUTINE
14
15         ;SET VOLUME VALID
16         ;FIRST, CLEAR -SET DIAGNOSTIC MODE-SYNCHRONIZE
17         ;PROM STROBE
18
19 023430   004737 023334          JSR      PC,CNTCLR      ;GO CLEAR CONTROLLER
20 023434   012760 000001 000024   MOV      #DMD,RMMR1(R0) ;LOAD RMMR1
21 023442   012704 000041          MOV      #33.,R4        ;ALLOW UP TO 33 BIT CLOCKS
22                                     ;TO SYNC PROM STROBE
23 023446   012760 005001 000024   5$:      MOV      #DMD!MUR!MCLK,RMMR1(R0) ;LOAD RMMR1
24 023454   012760 001001 000024   MOV      #DMD!MUR,RMMR1(R0) ;LOAD RMMR1
25 023462   016005 000024          MOV      RMMR1(R0),R5    ;STORE RMMR1 AT R5
26 023466   032705 000040          BIT      #WC,R5        ;WAIT FOR PROM STROBE TO COME ON
27 023472   001023          BNE     6$
28 023474   005304          DEC     R4
29 023476   001363          BNE     5$
30 023500   010037 001136          MOV      R0,$BDADR      ;PROM STROBE WONT SET
31 023504   062737 000024 001136   ADD     #RMMR1,$BDADR
32 023512   005037 001142          CLR     $BDDAT
33 023516   012737 000040 001140   MOV     #WC,$GDDAT
34 023524   062716 000002          ADD     #2,(SP)
35 023530   112776 000030 000000   MOV     #30,@(SP)
36 023536   000137 024300          JMP     60$            ;REPORT ERROR
37 023542   012760 005001 000024   6$:      MOV      #DMD!MUR!MCLK,RMMR1(R0) ;LOAD RMMR1
38 023550   012760 001001 000024   MOV      #DMD!MUR,RMMR1(R0) ;LOAD RMMR1
39 023556   016005 000024          MOV      RMMR1(R0),R5    ;STORE RMMR1 AT R5
40 023562   032705 000040          BIT      #WC,R5        ;WAIT FOR PROM STROBE TO GO OFF
41 023566   001423          BEQ     7$
42 023570   005304          DEC     R4
43 023572   001363          BNE     6$
44 023574   010037 001136          MOV      R0,$BDADR      ;PROM STROBE WONT RESET
45 023600   062737 000024 001136   ADD     #RMMR1,$BDADR
46 023606   012737 000040 001142   MOV     #WC,$BDDAT
47 023614   005037 001140          CLR     $GDDAT
48 023620   062716 000002          ADD     #2,(SP)
49 023624   112776 000062 000000   MOV     #62,@(SP)
50 023632   000137 024300          JMP     60$
51
52         ;SECOND, CLOCK INDEX PULSE TO
53         ; (1) CLEAR FORMAT CHANGE FLOP
54         ; (2) CLEAR RTC FLOP
55 023636   7$:
```



```

56 023636 012760 001005 000024      MOV      #DMD!MUR!MI,RMMR1(R0)      ;LOAD RMMR1
57 023644 012760 001001 000024      MOV      #DMD!MUR,RMMR1(R0)        ;LOAD RMMR1
58
59 023652 012760 000000 000014      MOV      #0,RMER1(R0)              ;LOAD RMER1
60 023660 012760 000000 000042      MOV      #0,RMER2(R0)              ;LOAD RMER2
61 023666 012760 000023 000000      MOV      #PACACK!GO,RMCS1(R0)      ;LOAD RMCS1
62 023674 016037 000012 001142      MOV      RMD5(R0),SBDDAT           ;STORE RMD5 AT SBDDAT
63 023702 042737 177677 001142      BIC      #^CVV,SBDDAT              ;VOLUME VALID SHOULD BE SET
64 023710 001021 000000 000000      BNE      10$
65 023712 010037 001136 000000      MOV      R0,$BDADR                 ;SETUP ERROR MSG
66 023716 062737 000012 001136      ADD      #RMD5,$BDADR
67 023724 012737 000100 001140      MOV      #VV,$GDDAT
68 023732 062716 000002 000000      ADD      #2,(SP)                   ;WRITE ERROR NUMBER
69 023736 112776 000100 000000      MOV      #100,@(SP)
70 023744 012737 000022 001174      MOV      #PACACK,$TMP0
71 023752 000552 000000 000000      BR       60$
72
73                                     ;LOAD REGISTERS
74 023754 013760 001414 000006      10$:   MOV      RMDAO,RMDA(R0)           ;LOAD RMDA
75 023754 013760 001442 000034      MOV      RMDCO,RMDC(R0)           ;LOAD RMDC
76 023762 013760 001440 000032      MOV      RMOFO,RMOF(R0)          ;LOAD RMOF
77 023770 013760 001410 000002      MOV      RMWCO,RMWC(R0)          ;LOAD RMWC
78 023776 013760 001412 000004      MOV      RMBAO,RMBA(R0)          ;LOAD RMBA
79 024004 013760 001412 000004
80
81                                     ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE
82 024012 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(R0) ;LOAD RMMR1
83 024020 012760 000000 000014      MOV      #0,RMER1(R0)            ;LOAD RMER1
84 024026 012760 000000 000042      MOV      #0,RMER2(R0)            ;LOAD RMER2
85 024034 013760 001406 000000      MOV      RMCS10,RMCS1(R0)        ;LOAD RMCS1
86
87                                     ;WAIT FOR 'RUN AND GO' TO SET
88 024042 012737 000310 001530      MOV      #200,WATCH               ;SET WATCHDOG TIMER VALUE
89 024050 004777 155456 001530      JSR      PC,@CLOCK                ;START THE CLOCK
90 024054 016037 000024 001142      20$:   MOV      RMMR1(R0),SBDDAT        ;STORE RMMR1 AT SBDDAT
91 024062 042737 137777 001142      BIC      #^CRG,SBDDAT
92 024070 001023 000000 000000      BNE      30$
93 024072 005737 001530 000000      TST      WATCH                    ;ANY TIME LEFT?
94 024076 001366 000000 000000      BNE      20$                       ;YES-WAIT
95 024100 004777 155430 000000      JSR      PC,@STOP                  ;STOP THE CLOCK
96 024104 012737 040000 001140      MOV      #RG,$GDDAT               ;SETUP ERROR MSG
97 024112 010037 001136 000000      MOV      R0,$BDADR
98 024116 062737 000024 001136      ADD      #RMMR1,$BDADR
99 024124 062716 000002 001136      ADD      #2,(SP)                   ;WRITE ERROR NUMBER IN
100 024130 112776 000101 000000      MOV      #101,@(SP)               ;USER'S CALL -
101 024136 000460 000000 000000      BR       60$
102 024140 004777 155370 000000      30$:   JSR      PC,@STOP                  ;STOP THE CLOCK
103
104                                     ;STEP COMMAND SEQUENCER TO HEAD SEQUENCE AT LOCATION 156 (17 CLOCKS)
105 024144 012704 000021 000000      40$:   MOV      #17.,R4                  ;R4=CLOCK COUNT
106 024150 012760 141401 000024      MOV      #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0) ;LOAD RMMR1
107 024156 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(R0) ;LOAD RMMR1
108 024164 005304 000000 000000      DEC      R4

```

```
109 024166 001370          BNE      40$
110
111          ;DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER TO PASS
112          ;TEST AT LOCATION 166
113 024170 012760 041001 000024      MOV      #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
114 024176 012760 041401 000024      MOV      #DMD!MUR!DBEN!MOC,RMMR1(R0) ;LOAD RMMR1
115
116          ;STEP SEQUENCER (35 CLOCKS) AND VERIFY SEARCH IS ENABLED
117 024204 012704 000043          MOV      #35.,R4 ;R4=CLOCK COUNT
118 024210          50$:
119 024210 012760 151401 000024      MOV      #DMD!MUR!DBEN!MOC!DBCK!DTO,RMMR1(R0) ;LOAD RMMR1
120 024216 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
121 024224 005304          DEC      R4
122 024226 001370          BNE      50$
123 024230 016037 000024 001142      MOV      RMMR1(R0),SBDDAT ;STORE RMMR1 AT SBDDAT
124 024236 042737 173777 001142      BIC      #^CESRC,SBDDAT ;SEARCH SHOULD BE ENABLED
125 024244 001015          BNE      60$
126 024246 010037 001136          MOV      R0,SBDADR ;SETUP ERROR MESSAGE
127 024252 062737 000024 001136      ADD      #RMMR1,SBDADR
128 024260 012737 004000 001140      MOV      #ESRC,$GDDAT
129 024266 062716 000002          ADD      #2,(SP) ;WRITE ERROR NUMBER
130 024272 112776 000102 000000      MOVB    #102,@(SP)
131 024300          60$:
132 024300 000207          RTS      PC
133
```

```
1      .SBTTL  SECTOR COMPARE SUBROUTINE
2
3      ;THIS SUBROUTINE CONTINUES THE EXECUTION OF A DATA COMMAND
4      ;FROM WHERE SEARCH HAS BEEN ENABLED TO WHERE SECTOR COMPARE
5      ;IS SET.
6
7      024302      SCTCMP:
8
9      ;SET INDEX PULSE TO CLEAR FORMAT CHANGE FLOP
10     024302      012760      051405      000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MI,RMMR1(R0)      ;LOAD RMMR1
11     024310      012760      051401      000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12
13     ;WAIT AT LEAST 4 MS FOR 'RETURN TO CENTER LINE' ONE SHOT TO SET
14     024316      012737      000006      001530      MOV      #6,WATCH      ;SET WATCHDOG TIMER VALUE
15     024324      004777      155202      JSR      PC,@CLOCK      ;START THE CLOCK
16     024330      005737      001530      10$:      TST      WATCH
17     024334      001375      BNE      10$
18     024336      004777      155172      JSR      PC,@STOP      ;STOP THE CLOCK
19
20     ;DATA INPUT TO SEARCH ENABLE FLOP SHOULD BE ONE - CLOCK SECTOR PULSE TO
21     ;SET FLOP
22     024342      012760      051441      000024      MOV      #DMD!MUR!DBEN!MOC!DTU!MS,RMMR1(R0) ;LOAD RMMR1
23     024350      012760      051401      000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
24
25     ;WITH SECTOR COMPARE HIGH, CLOCK SECTOR PULSE TO SET SECTOR COMPARE FLOP
26     024356      012760      051403      000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
27     024364      012760      051443      000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MSC!MS,RMMR1(R0) ;LOAD RMMR1
28     024372      012760      051403      000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MSC,RMMR1(R0) ;LOAD RMMR1
29     024400      012760      051401      000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
30     024406      000207      RTS      PC
31
```

```

1      .SBTTL SET LOOKING FOR SYNC SUBROUTINE
2
3      ;THIS SUBROUTINE WILL SETUP THE DATA TIMING SEQUENCER
4      ;ASSUMING SEARCH IS ENABLED,TO THE POINT WHERE 'PLFS' IS ACTIVE
5
6 024410 SETLFS:
7
8      ;PULSE BIT CLOCK UNTIL PROM STROBE SETS
9 024410 10$:
10 024410 012760 055401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0)      ;LOAD RMMR1
11 024416 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12 024424 016005 000024      MOV      RMMR1(R0),R5      ;STORE RMMR1 AT R5
13 024430 032705 000040      BIT      #WC,R5
14 024434 001765      BEQ      10$
15
16      ;SET UP DATA SEQUENCER TO LOCATION 11. WHERE PLFS WILL SET
17 024436 012704 000260      20$:
18 024442      MOV      #176.,R4      ;MAX NUMBER OF BIT CLOCK
19 024442 012760 055401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO!MCLK,RMMR1(R0) ;LOAD RMMR1
20 024450 012760 051401 000024      MOV      #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
21 024456 016005 000024      MOV      RMMR1(R0),R5      ;STORE RMMR1 AT R5
22 024462 032705 002000      BIT      #PLFS,R5      ;EXIT IF PLFS IS SET
23 024466 001010      BNE      30$
24 024470 005304      DEC      R4      ;ERROR IF COUNT EXHAUSTED
25 024472 001363      BNE      20$
26 024474 062716 000002      ADD      #2,(SP)      ;
27 024500 112776 000035 000000      MOV      #35,@(SP)      ;CANT SET PLFS
28 024506 000430      BR      50$
29
30      ;STEP THE MAIN REG CLOCK UNTIL THE TRAILING EDGE OF PROM STORBE
31      ;IS DETECTED.
32 024510 30$:
33 024510 016005 000024      MOV      RMMR1(R0),R5      ;STORE RMMR1 AT R5
34 024514 032705 000040      BIT      #WC,R5
35 024520 001007      BNE      40$
36 024522 012760 055401 000024      MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
37 024530 012760 051401 000024      MOV      #MR1AAA,RMMR1(R0) ;LOAD RMMR1
38 024536 000414      BR      50$
39 024540 40$:
40 024540 016005 000024      MOV      RMMR1(R0),R5      ;STORE RMMR1 AT R5
41 024544 032705 000040      BIT      #WC,R5
42 024550 001407      BEQ      50$
43 024552 012760 055401 000024      MOV      #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
44 024560 012760 051401 000024      MOV      #MR1AAA,RMMR1(R0) ;LOAD RMMR1
45 024566 000764      BR      40$
46 024570 50$:
47 024570 000207      RTS      PC      ;EXIT

```

```

1          .SBTTL  CLOCK SYNC SUBROUTINE
2
3          ;THIS SUBROUTINE WILL SIMULATE THE HEADER AND DATA SYNC
4          ;PATTERN BEING READ
5
6 024572   CLKSNC:
7
8          ;VERIFY THAT 'PLFS' IS ON
9 024572   016004 000024   MOV     RMMR1(R0),R4   ;STORE RMMR1 AT R4
10 024576  032704 002000   BIT     #PLFS,R4      ;LOOK FOR SYN SET ?
11 024602  001023          BNE     10$           ;BRANCH IF SO
12 024604  010437 001142   MOV     R4,$BDDAT    ;
13 024610  042737 175777 001142   BIC     #^CPLFS,$BDDAT
14 024616  012737 002000 001140   MOV     #PLFS,$GDDAT
15 024624  010037 001136   MOV     R0,$BDADR
16 024630  062737 000024 001136   ADD     #RMMR1,$BDADR
17 024636  062716 000002   ADD     #2,(SP)
18 024642  112776 000035 000000   MOV     #35,@(SP)
19 024650  000472          BR      60$
20 024652  012705 000021   10$:   MOV     #17.,R5      ;MAKE SURE PROM STROBE IS OFF
21 024656  032704 000040   20$:   BIT     #WC,R4
22 024662  001434          BEQ     30$
23 024664  012760 055401 000024   MOV     #MR1AAA!MCLK,RMMR1(R0) ;LOAD RMMR1
24 024672  012760 051401 000024   MOV     #MR1AAA,RMMR1(R0)      ;LOAD RMMR1
25 024700  016004 000024   MOV     RMMR1(R0),R4          ;STORE RMMR1 AT R4
26 024704  005305          DEC     R5
27 024706  001363          BNE     20$
28
29          ;ERROR CAN'T RESET PROM STROBE WITH LFS ACTIVE
30 024710  010437 001142   MOV     R4,$BDDAT          ;SETUP ERROR FOR USER
31 024714  042737 177737 001142   BIC     #^CWC,$BDDAT
32 024722  005037 001140   CLR     $GDDAT
33 024726  010037 001136   MOV     R0,$BDADR
34 024732  062737 000024 001136   ADD     #RMMR1,$BDADR
35 024740  062716 000002   ADD     #2,(SP)
36 024744  112776 000062 000000   MOV     #62,@(SP)
37 024752  000431          BR      60$
38
39          ;CLOCK THE SYNC PATTERN (00011001) THROUGH THE SHIFT REGISTER
40 024754   30$:
41 024754  012704 014400   MOV     #014400,R4
42 024760  012737 000020 025040   MOV     #16.,70$
43 024766  012705 051401   40$:   MOV     #MR1AAA,R5      ;STROBE BIT COUNT
44 024772  000241          CLC
45 024774  006004          ROR     R4              ;GENERATE REG VALUE
46 024776  103002          BCC     50$            ;WITH MAINTENANCE CLOCK ON
47 025000  052705 002000   BIS     #MRD,R5        ;SET READ BIT PER PATTERN BIT
48
49 025004   50$:
50 025004  010560 000024   MOV     R5,RMMR1(R0)      ;LOAD RMMR1
51 025010  052705 004000   BIS     #MCLK,R5
52 025014  010560 000024   MOV     R5,RMMR1(R0)      ;LOAD RMMR1
53 025020  042705 004000   BIC     #MCLK,R5          ;CLOCK ONE BIT
54 025024  010560 000024   MOV     R5,RMMR1(R0)      ;LOAD RMMR1
55 025030  005337 025040   DEC     70$
56 025034  001354          BNE     40$

```

57
58
59
60 025036
61 025036 000207
62
63 025040 000000
64

:CAN'T VERIFY SYNC CLOCK WAS DETECTED
:USER CAN DO SO BY SLEEPING
:BIT CLOCK AND VERIFY PROM STROBE SETS WITHIN ONE WORD TIME
60\$: RTS PC
70\$: .word 0 :TEMPORARY STORAGE

1

.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
    
```

```

025042
025042 010046
025044 010146
025046 010246
025050 010346
025052 010446
025054 010546
025056 016646 000022
025062 016646 000022
025066 016646 000022
025072 016646 000022
025076 000002
    
```

```

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

```

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

```

025100
025100 012666 000022
025104 012666 000022
025110 012666 000022
025114 012666 000022
025120 012505
025122 012604
025124 012603
025126 012602
025130 012601
025132 012600
025134 000002
    
```

```

$RESREG:
MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTI
    
```

2

.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
    
```

```

025136 010146
025140 016601 000006
025144 000261
    
```

```

$TYPBN: MOV R1,-(SP) ;;SAVE R1 ON THE STACK
MOV 6(SP),R1 ;;GET THE INPUT NUMBER
SEC ;;SET 'C' SO CAN KEEP TRACK OF THE NUMBER OF BITS
    
```

```

025146 112737 000060 025210 1$:  MOVB  #'0,$BIN      ;;SET CHARACTER TO AN ASCII '0'.
025154 006101          ROL  R1          ;;GET THIS BIT
025156 001406          BEQ  2$          ;;DONE?
025160 105537 025210  ADCB  $BIN          ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
025164 104401 025210  TYPE  , $BIN        ;;GO TYPE THIS BIT
025170 000241          CLC          ;;CLEAR 'C' SO CAN KEEP TRACK OF BITS
025172 000765          BR   1$          ;;GO DO THE NEXT BIT
025174 012601          MOV  (SP)+,R1      ;;POP THE STACK INTO R1
025176 016666 000002 000004 2$:  MOV  2(SP),4(SP)    ;;ADJUST THE STACK
025204 012616          MOV  (SP)+,(SP)
025206 000002          RTI          ;;RETURN TO USER
025210 000 000  $BIN:  .BYTE  0,0      ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
3  .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

```

```

025212 010046          $TYPDS:  MOV  R0,-(SP)      ;;PUSH R0 ON STACK
025212 010146          MOV  R1,-(SP)      ;;PUSH R1 ON STACK
025216 010246          MOV  R2,-(SP)      ;;PUSH R2 ON STACK
025220 010346          MOV  R3,-(SP)      ;;PUSH R3 ON STACK
025222 010546          MOV  R5,-(SP)      ;;PUSH R5 ON STACK
025224 012746 020200  MOV  #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
025230 016605 000020  MOV  20(SP),R5      ;;GET THE INPUT NUMBER
025234 100004          BPL  1$          ;;BR IF INPUT IS POS.
025236 005405          NEG  R5          ;;MAKE THE BINARY NUMBER POS.
025240 112766 000055 000001 1$:  MOVB #'-',1(SP)    ;;MAKE THE ASCII NUMBER NEG.
025246 005000          CLR  R0          ;;ZERO THE CONSTANTS INDEX
025250 012703 025426  MOV  #SDBLK,R3      ;;SETUP THE OUTPUT POINTER
025254 112723 000040  MOVB #'',(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
025260 005002          CLR  R2          ;;CLEAR THE BCD NUMBER
025262 016001 025416  MOV  $DTBL(R0),R1   ;;GET THE CONSTANT
025266 160105          3$:  SUB  R1,R5      ;;FORM THIS BCD DIGIT
025270 002402          BLT  4$          ;;BR IF DONE
025272 005202          INC  R2          ;;INCREASE THE BCD DIGIT BY 1
025274 000774          BR   3$
025276 040105          4$:  ADD  R1,R5      ;;ADD BACK THE CONSTANT
025300 005702          TST  R2          ;;CHECK IF BCD DIGIT=0
025302 001002          BNE  5$          ;;FALL THROUGH IF 0
025304 105716          TSTB (SP)        ;;STILL DOING LEADING 0'S?
025306 100407          BMI  7$          ;;BR IF YES
025310 106316          5$:  ASLB (SP)        ;;MSD?
025312 103003          BCC  6$          ;;BR IF NO
025314 116663 000001 177777  MOVB 1(SP),-1(R3)  ;;YES--SET THE SIGN
025322 052702 000060  6$:  BIS  #'0,R2      ;;MAKE THE BCD DIGIT ASCII
025326 052702 000040  7$:  BIS  #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
025332 110223          MOVB R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
025334 005720          TST  (R0)+     ;;JUST INCREMENTING
025336 020027 000010  CMP  R0,#10      ;;CHECK THE TABLE INDEX

```



```

025342 002746          BLT      2$          ;;GO DO THE NEXT DIGIT
025344 003002          BGT      8$          ;;GO TO EXIT
025346 010502          MOV      R5,R2        ;;GET THE LSD
025350 000764          BR       6$          ;;GO CHANGE TO ASCII
025352 105726          8$:    TSTB    (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
025354 100003          BPL      9$          ;;BR IF NO
025356 116663 177777 177776  MOVB    -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
025364 105013          9$:    CLRB    (R3)          ;;SET THE TERMINATOR
025366 012605          MOV      (SP)+,R5      ;;POP STACK INTO R5
025370 012603          MOV      (SP)+,R3      ;;POP STACK INTO R3
025372 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
025374 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
025376 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
025400 104401 025426    TYPE     $DBLK          ;;NOW TYPE THE NUMBER
025404 016666 000002 000004  MOV      2(SP),4(SP)   ;;ADJUST THE STACK
025412 012616          MOV      (SP)+,(SP)
025414 000002          RTI
025416 023420          $DTBL: 10000.        ;;RETURN TO USER
025420 001750          1000.
025422 000144          100.
025424 000012          10.
025426          $DBLK: .BLKW 4
          .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
  
```

4

```

*****
*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
  
```

```

025436 017646 000000          $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
025442 116637 000001 025661  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
025450 112637 025663          MOVB    (SP)+,$OMODE+1 ;;NUMBER OF DIGITS TO TYPE
025454 062716 000002          ADD     #2,(SP)        ;;ADJUST RETURN ADDRESS
025460 000406          BR      $TYPON
025462 112737 000001 025661  $TYPOC: MOVB    #1,$OFILL    ;;SET THE ZERO FILL SWITCH
025470 112737 000006 025663  MOVB    #6,$OMODE+1    ;;SET FOR SIX(6) DIGITS
025476 112737 000005 025660  $TYPON: MOVB    #5,$OCNT    ;;SET THE ITERATION COUNT
025504 010346          MOV     R3,-(SP)      ;;SAVE R3
025506 010446          MOV     R4,-(SP)      ;;SAVE R4
  
```

```

025510 010546          MOV      R5,-(SP)      ;;SAVE R5
025512 113704 025663  MOVVB   $OMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
025516 005404          NEG      R4
025520 062704 000006  ADD      #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
025524 110437 025662  MOVVB   R4,$OMODE     ;;SAVE IT FOR USE
025530 113704 025661  MOVVB   $OFILL,R4     ;;GET THE ZERO FILL SWITCH
025534 016605 000012  MOV      12(SP),R5    ;;PICKUP THE INPUT NUMBER
025540 005003          CLR      R3          ;;CLEAR THE OUTPUT WORD
025542 006105          1$:    ROL      R5          ;;ROTATE MSB INTO 'C'
025544 000404          BR       3$          ;;GO DO MSB
025546 006105          2$:    ROL      R5          ;;FORM THIS DIGIT
025550 006105          ROL      R5
025552 006105          ROL      R5
025554 010503          MOV      R5,R3
025556 006103          3$:    ROL      R3          ;;GET LSB OF THIS DIGIT
025560 105337 025662  DECB   $OMODE        ;;TYPE THIS DIGIT?
025564 100016          BPL     7$          ;;BR IF NO
025566 042703 177770  BIC     #177770,R3   ;;GET RID OF JUNK
025572 001002          BNE     4$          ;;TEST FOR 0
025574 005704          TST     R4          ;;SUPPRESS THIS 0?
025576 001403          BEQ     5$          ;;BR IF YES
025600 005204          4$:    INC     R4          ;;DON'T SUPPRESS ANYMORE 0'S
025602 052703 000060  BIS     #'0,R3       ;;MAKE THIS DIGIT ASCII
025606 052703 000040  5$:    BIS     #' ,R3   ;;MAKE ASCII IF NOT ALREADY
025612 110337 025656  MOVVB   R3,8$        ;;SAVE FOR TYPING
025616 104401 025656  TYPE    ,8$          ;;GO TYPE THIS DIGIT
025622 105337 025660  7$:    DECB   $OCNT    ;;COUNT BY 1
025626 003347          BGT     2$          ;;BR IF MORE TO DO
025630 002402          BLT     6$          ;;BR IF DONE
025632 005204          INC     R4          ;;INSURE LAST DIGIT ISN'T A BLANK
025634 000744          BR      2$          ;;GO DO THE LAST DIGIT
025636 012605          6$:    MOV     (SP)+,R5   ;;RESTORE R5
025640 012604          MOV     (SP)+,R4   ;;RESTORE R4
025642 012603          MOV     (SP)+,R3   ;;RESTORE R3
025644 016666 000002 000004  MOV     2(SP),4(SP) ;;SET THE STACK FOR RETURNING
025652 012616          MOV     (SP)+,(SP)
025654 000002          RTI          ;;RETURN
025656 000          8$:    .BYTE  0          ;;STORAGE FOR ASCII DIGIT
025657 000          .BYTE  0          ;;TERMINATOR FOR TYPE ROUTINE
025660 000          $OCNT: .BYTE  0          ;;OCTAL DIGIT COUNTER
025661 000          $OFILL: .BYTE  0          ;;ZERO FILL SWITCH
025662 000000          $OMODE: .WORD  0          ;;NUMBER OF DIGITS TO TYPE
          .SBTTL TYPE ROUTINE
    
```

5

```

;*****
;*ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;*
;*CALL:
;*1) USING A TRAP INSTRUCTION
;*   TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCII STRING
;*OR
;*   TYPE
;*   MESADR
    
```

```

; *
025664 105737 001173 $TYPE: TSTB $TPFLG :: IS THERE A TERMINAL?
025670 100002 BPL 1$ :: BR IF YES
025672 000000 HALT :: HALT HERE IF NO TERMINAL
025674 000430 BR 3$ :: LEAVE
025676 010046 1$: MOV R0,-(SP) :: SAVE R0
025700 017600 000002 MOV @2(SP),R0 :: GET ADDRESS OF ASCIZ STRING
025704 122737 000001 001242 CMPB #APTENV,$ENV :: RUNNING IN APT MODE
025712 001011 BNE 62$ :: NO, GO CHECK FOR APT CONSOLE
025714 132737 000100 001243 BITB #APTPOOL,$ENVM :: SPOOL MESSAGE TO APT
025722 001405 BEQ 62$ :: NO, GO CHECK FOR CONSOLE
025724 010037 025734 MOV R0,61$ :: SETUP MESSAGE ADDRESS FOR APT
025730 004737 030634 JSR PC,$ATY3 :: SPOOL MESSAGE TO APT
025734 000000 61$: .WORD 0 :: MESSAGE ADDRESS
025736 132737 000040 001243 62$: BITB #APTCSUP,$ENVM :: APT CONSOLE SUPPRESSED
025744 001003 BNE 60$ :: YES, SKIP TYPE OUT
025746 112046 2$: MOV (R0)+,-(SP) :: PUSH CHARACTER TO BE TYPED ONTO STACK
025750 001005 BNE 4$ :: BR IF IT ISN'T THE TERMINATOR
025752 005726 TST (SP)+ :: IF TERMINATOR POP IT OFF THE STACK
025754 012600 60$: MOV (SP)+,R0 :: RESTORE R0
025756 062716 000002 3$: ADD #2,(SP) :: ADJUST RETURN PC
025762 000002 RTI :: RETURN
025764 122716 000011 4$: CMPB #HT,(SP) :: BRANCH IF <HT>
025770 001430 BEQ 8$
025772 122716 000200 CMPB #CRLF,(SP) :: BRANCH IF NOT <CRLF>
025776 001006 BNE 5$
026000 005726 TST (SP)+ :: POP <CR><LF> EQUIV
026002 104401 TYPE :: TYPE A CR AND LF
026004 001217 $CRLF
026006 105037 026214 CLR B $CHARCNT :: CLEAR CHARACTER COUNT
026012 000755 BR 2$ :: GET NEXT CHARACTER
026014 004737 026076 5$: JSR PC,$TYPEC :: GO TYPE THIS CHARACTER
026020 123726 001172 6$: CMPB $FILLC,(SP)+ :: IS IT TIME FOR FILLER CHARS.?
026024 001350 BNE 2$ :: IF NO GO GET NEXT CHAR.
026026 013746 001170 MOV $NULL,-(SP) :: GET # OF FILLER CHARS. NEEDED
:: AND THE NULL CHAR.
026032 105366 000001 7$: DECB 1(SP) :: DOES A NULL NEED TO BE TYPED?
026036 002770 BLT 6$ :: BR IF NO--GO POP THE NULL OFF OF STACK
026040 004737 026076 JSR PC,$TYPEC :: GO TYPE A NULL
026044 105337 026214 DECB $CHARCNT :: DO NOT COUNT AS A COUNT
026050 000770 BR 7$ :: LOOP

```

;HORIZONTAL TAB PROCESSOR

```

026052 112716 000040 8$: MOV #',(SP) :: REPLACE TAB WITH SPACE
026056 004737 026076 9$: JSR PC,$TYPEC :: TYPE A SPACE
026062 132737 000007 026214 BITB #7,$CHARCNT :: BRANCH IF NOT AT
026070 001372 BNE 9$ :: TAB STOP
026072 005726 TST (SP)+ :: POP SPACE OFF STACK
026074 000724 BR 2$ :: GET NEXT CHARACTER
026076 $TYPEC:
026076 105777 153056 TSTB @TKS :: CHAR IN KYBD BUFFER?
026102 100022 BPL 10$ :: BR IF NOT
026104 017746 153052 MOV @TKB,-(SP) :: GET CHAR
026110 042716 177600 BIC #177600,(SP) :: STRIP EXTRANEIOUS BITS
026114 122716 000023 CMPB #$XOFF,(SP) :: WAS CHAR XOFF

```

```

026120 001012          BNE      102$          ;;BR IF NOT
026122          101$:  TSTB      @STKS          ;;WAIT FOR CHAR
026122 105777 153032  BPL      101$
026126 100375          MOVB      @STKB,(SP)      ;;GET CHAR
026130 117716 153026  BIC      #177600,(SP)      ;;STRIP IT
026134 042716 177600  CMPB      #$XON,(SP)      ;;WAS IT XON?
026140 122716 000021  BNE      101$          ;;BR IF NOT
026144 001366          102$:  TST      (SP)+          ;;FIX STACK
026146 005726          10$:  TSTB      @STPS          ;;WAIT UNTIL PRINTER IS READY
026150          BPL      10$
026150 105777 153010  MOVB      2(SP),@STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
026154 100375          CMPB      #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
026156 116677 000002 153002  BNE      1$          ;;BRANCH IF NO
026164 122766 000015 000002  CLRB      $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
026172 001003          BR       $TYPEX      ;;EXIT
026174 105037 026214  CMPB      #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
026200 000406          BEQ      $TYPEX      ;;BRANCH IF YES
026202 122766 000012 000002 1$:  INCB      (PC)+          ;;COUNT THE CHARACTER
026210 001402          $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
026212 105227          $TYPEX: RTS      PC
026214 000000
026216 000207

```

6

.SBTTL SCOPE HANDLER ROUTINE

```

;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG ($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

```

```

026220          $SCOPE:
026220 104410          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
026222 032777 040000 152724 1$:  BIT      #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
026230 001131          BNE      $OVER          ;;YES IF SW14=1
026232 000416          ;####START OF CODE FOR THE XOR TESTER####
          $XTSTR: BR      6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
          ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
026234 013746 000004          MOV      @ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
026240 012737 026260 000004  MOV      #5$,@ERRVEC      ;;SET FOR TIMEOUT
026246 005737 177060          TST      @177060          ;;TIME OUT ON XOR?
026252 012637 000004          MOV      (SP)+,@ERRVEC      ;;RESTORE THE ERROR VECTOR
026256 000500          BR       $SVLAD          ;;GO TO THE NEXT TEST
026260 022626          5$:  CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
026262 012637 000004          MOV      (SP)+,@ERRVEC      ;;RESTORE THE ERROR VECTOR
026266 000440          BR       7$          ;;LOOP ON THE PRESENT TEST
026270          6$:;####END OF CODE FOR THE XOR TESTER####
026270 032777 000400 152656  BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
026276 001421          BEQ      2$          ;;BR IF NO
026300 005046          CLR      -(SP)          ;;CLEAR A TEMP. LOCATION

```

```

SCOPE HANDLER ROUTINE

026302 117716 152646      MOVB   @SWR,(SP)      ;;PICKUP THE DESIRED TEST NUMBER
026306 001414              BEQ    8$             ;;BRANCH IF BAD TEST NUMBER IN SWR
026310 022716 000024      CMP    #24,(SP)      ;;CHECK THE NUMBER IN THE SWR
026314 002411              BLT    8$             ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
026316 011637 001116      MOV    (SP), $STSTNM ;;UPDATE THE TEST NUMBER
026322 005316              DEC    (SP)          ;;BACKUP BY ONE
026324 006316              ASL    (SP)          ;;SCALE THE TEST NUMBER AS AN INDEX
026326 062716 026532      ADD    $$SW08TBL,(SP) ;;FORM THE ADDRESS OF TEST POINTER
026332 013637 001122      MOV    @($SP)+,$LPADR ;;SET LOOP ADDRESS TO DESIRED TEST
026336 000466              BR     $OVER         ;;GO LOOP ON THE TEST
026340 005726              8$: TST   (SP)+         ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
026342 105737 001117      2$: TSTB  $ERFLG       ;;HAS AN ERROR OCCURRED?
026346 001421              BEQ    3$             ;;BR IF NO
026350 123737 001131 001117      CMPB  $ERMAX,$ERFLG  ;;MAX. ERRORS FOR THIS TEST OCCURRED?
026356 101015              BHI    3$             ;;BR IF NO
026360 032777 001000 152566      BIT   #BIT09,@SWR    ;;LOOP ON ERROR?
026366 001404              BEQ    4$             ;;BR IF NO
026370 013737 001124 001122      7$: MOV  $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
026376 000446              BR     $OVER
026400 105037 001117      4$: CLRB  $ERFLG       ;;ZERO THE ERROR FLAG
026404 005037 001206      CLR   $TIMES         ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
026410 000415              BR     1$            ;;ESCAPE TO THE NEXT TEST
026412 032777 004000 152534      3$: BIT   #BIT11,@SWR  ;;INHIBIT ITERATIONS?
026420 001011              BNE    1$            ;;BR IF YES
026422 005737 001230      TST   $PASS         ;;IF FIRST PASS OF PROGRAM
026426 001406              BEQ    1$            ;;INHIBIT ITERATIONS
026430 005237 001120      INC   $ICNT         ;;INCREMENT ITERATION COUNT
026434 023737 001206 001120      CMP   $TIMES,$ICNT  ;;CHECK THE NUMBER OF ITERATIONS MADE
026442 002024              BGE    $OVER        ;;BR IF MORE ITERATION REQUIRED
026444 012737 000001 001120      1$: MOV  #1,$ICNT     ;;REINITIALIZE THE ITERATION COUNTER
026452 013737 026530 001206      MOV   $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
026460 105237 001116      $SVLAD: INCB  $STSTNM    ;;COUNT TEST NUMBERS
026464 113737 001116 001226      MOVB  $STSTNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
026472 011637 001122      MOV   (SP),$LPADR   ;;SAVE SCOPE LOOP ADDRESS
026476 011637 001124      MOV   (SP),$LPERR   ;;SAVE ERROR LOOP ADDRESS
026502 005037 001210      CLR   $ESCAPE       ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
026506 112737 000001 001131      MOVB  #1,$ERMAX     ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
026514 013777 001116 152434      $OVER: MOV  $STSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
026522 013716 001122      MOV   $LPADR,(SP)  ;;FUDGE RETURN ADDRESS
026526 000002              RTI                ;;FIXES PS
026530 000024              $MXCNT: 20.        ;;MAX. NUMBER OF ITERATIONS
026532 000024              $SW08TBL:
026532 005400              .WORD  TST1+2      ;;STARTING ADDRESS OF TEST 1
026534 005710              .WORD  TST2+2      ;;STARTING ADDRESS OF TEST 2
026536 006072              .WORD  TST3+2      ;;STARTING ADDRESS OF TEST 3
026540 006242              .WORD  TST4+2      ;;STARTING ADDRESS OF TEST 4
026542 006372              .WORD  TST5+2      ;;STARTING ADDRESS OF TEST 5
026544 007456              .WORD  TST6+2      ;;STARTING ADDRESS OF TEST 6
026546 010526              .WORD  TST7+2      ;;STARTING ADDRESS OF TEST 7
026550 010664              .WORD  TST10+2     ;;STARTING ADDRESS OF TEST 10
026552 010750              .WORD  TST11+2     ;;STARTING ADDRESS OF TEST 11
026554 011220              .WORD  TST12+2     ;;STARTING ADDRESS OF TEST 12
026556 011550              .WORD  TST13+2     ;;STARTING ADDRESS OF TEST 13
026560 012134              .WORD  TST14+2     ;;STARTING ADDRESS OF TEST 14
026562 012462              .WORD  TST15+2     ;;STARTING ADDRESS OF TEST 15
026564 013132              .WORD  TST16+2     ;;STARTING ADDRESS OF TEST 16
026566 013556              .WORD  TST17+2     ;;STARTING ADDRESS OF TEST 17

```

```

026570 014000 .WORD TST20+2 ;; STARTING ADDRESS OF TEST 20
026572 014472 .WORD TST21+2 ;; STARTING ADDRESS OF TEST 21
026574 015460 .WORD TST22+2 ;; STARTING ADDRESS OF TEST 22
026576 016726 .WORD TST23+2 ;; STARTING ADDRESS OF TEST 23
026600 020116 .WORD TST24+2 ;; STARTING ADDRESS OF TEST 24
7
    
```

.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
    
```

```

026602 $ERROR:
026602 104410 CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
026604 105237 001117 7$: INCB $ERFLG ;; SET THE ERROR FLAG
026610 001775 BEQ 7$ ;; DON'T LET THE FLAG GO TO ZERO
026612 013777 001116 152336 MOV $TSTNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
026620 032777 002000 152326 BIT #BIT10,@SWR ;; BELL ON ERROR?
026626 001402 BEQ 1$ ;; NO - SKIP
026630 104401 001212 TYPE ,SBELL ;; RING BELL
026634 005237 001126 1$: INC $ERTTL ;; COUNT THE NUMBER OF ERRORS
026640 011637 001132 MOV (SP), $ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
026644 162737 000002 001132 SUB #2,$ERRPC
026652 117737 152254 001130 MOV @ERRPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
026660 032777 020000 152266 BIT #BIT13,@SWR ;; SKIP TYPEOUT IF SET
026666 001004 BNE 20$ ;; SKIP TYPEOUTS
026670 004737 021566 JSR PC,ERRTP ;; GO TO USER ERROR ROUTINE
026674 104401 001217 TYPE ,SCRLF
026700 20$: CMPB #APTENV,$ENV ;; RUNNING IN APT MODE
026700 122737 000001 001242 BNE 2$ ;; NO, SKIP APT ERROR REPORT
026706 001007 MOV @ITEMB,21$ ;; SET ITEM NUMBER AS ERROR NUMBER
026710 113737 001130 026722 JSR PC,$ATY4 ;; REPORT FATAL ERROR TO APT
026716 004737 030644
026722 000 .BYTE 0 21$: .BYTE 0
026723 000 .BYTE 0
026724 000777 22$: BR 22$ ;; APT ERROR LOOP
026726 005777 152222 2$: TST @SWR ;; HALT ON ERROR
026732 100002 BPL 3$ ;; SKIP IF CONTINUE
026734 000000 HALT ;; HALT ON ERROR!
026736 104410 CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
026740 032777 001000 152206 3$: BIT #BIT09,@SWR ;; LOOP ON ERROR SWITCH SET?
026746 001402 BEQ 4$ ;; BR IF NO
026750 013716 001124 MOV $LPERR,(SP) ;; FUDGE RETURN FOR LOOPING
026754 005737 001210 4$: TST $ESCAPE ;; CHECK FOR AN ESCAPE ADDRESS
026760 001402 BEQ 5$ ;; BR IF NONE
026762 013716 001210 MOV $ESCAPE,(SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE
026766 5$:
026766 022737 021546 000042 CMP #SENDAD,@#42 ;; ACT-11 AUTO-ACCEPT?
026774 001001 BNE 6$ ;; BRANCH IF NO
026776 000000 HALT ;; YES
    
```

027000
027000 000002
8

6\$: RTI ;:RETURN
.SBTTL TTY INPUT ROUTINE

027002 000000
027004 000000
027006 000000
027010 027011

.ENABL LSB
\$TKCNT: .WORD 0 ;:NUMBER OF ITEMS IN QUEUE
\$TKQIN: .WORD 0 ;:INPUT POINTER
\$TKQOUT: .WORD 0 ;:OUTPUT POINTER
\$TKQSRT: .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

027012 005037 027002
027016 012737 027010 027004
027024 013737 027004 027006
027032 012737 027062 000060
027040 012737 000200 000062
027046 005777 152110
027052 012777 000100 152100
027060 000207

\$TKINT: CLR \$TKCNT ;:CLEAR COUNT OF ITEMS IN QUEUE
MOV #\$TKQSRT,\$TKQIN ;:MOVE THE STARTING ADDRESS OF THE
MOV \$TKQIN,\$TKQOUT ;:QUEUE INTO THE INPUT & OUTPUT POINTERS.
MOV #\$TKSRV,@\$TKVEC ;:INITIALIZE THE KEYBOARD VECTOR
MOV #200,@\$TKVEC+2 ;:'BR' LEVEL 4
TST @\$TKB ;:CLEAR DONE FLAG
MOV #100,\$TKS ;:ENABLE TTY KEYBOARD INTERRUPT
RTS PC ;:RETURN TO CALLER

;*TK SERVICE ROUTINE
;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
;*IT IN THE QUEUE.
;*IF THE CHARACTER IS A 'CONTROL-C' (^C) \$TKINT IS CALLED AND
;*UPON RETURN EXIT IS MADE TO THE 'CONTROL-C' RESTART ADDRESS (SHUT)

027062 117746 152074
027066 042716 177600
027072 021627 000003
027076 001007
027100 104401 030176
027104 004737 027012
027110 005726
027112 000137 023362
027116 021627 000007
027122 001004
027124 022737 000176 001154
027132 001500

\$TKSRV: MOVB @\$TKB,-(SP) ;:PICKUP THE CHARACTER
BIC #^C177,(SP) ;:STRIP THE JUNK
CMP (SP),#3 ;:IS IT A CONTROL C?
BNE 1\$;:BRANCH IF NO
TYPE ,SCNTLC ;:TYPE A CONTROL-C (^C)
JSR PC,\$TKINT ;:INIT THE KEYBOARD
TST (SP)+ ;:CLEAN UP STACK
JMP SHUT ;:CONTROL C RESTART
1\$: CMP (SP),#7 ;:IS IT A CONTROL G?
BNE 2\$;:BRANCH IF NO
CMP #SWREG,SWR ;:IS SOFT-SWR SELECTED?
BEQ 6\$;:GO TO SWR CHANGE

027134
027134 022737 000001 027002
027142 001004
027144 104401 001212
027150 005726
027152 000451
027154 021627 000023

2\$: CMP #1,\$TKCNT ;:IS THE QUEUE FULL?
BNE 3\$;:BRANCH IF NO
TYPE ,SBELL ;:RING THE TTY BELL
TST (SP)+ ;:CLEAN CHARACTER OFF OF STACK
BR 5\$;:EXIT
3\$: CMP (SP),#25 ;:IS IT A CONTROL-S?

```

027160 001021      BNE      32$      ;;BRANCH IF NO
027162 005077 151772 CLR      @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
027166 005726      TST      (SP)+      ;;CLEAN CHAR OFF STACK
027170 105777 151764 31$:  TSTB     @STKS      ;;WAIT FOR A CHAR
027174 100375      BPL      31$      ;;LOOP UNTIL ITS THERE
027176 117746 151760 MOVB     @STKB,-(SP) ;;GET THE CHARACTER
027202 042716 177600 BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
027206 022627 000021 CMP      (SP)+,#21   ;;IS IT A CONTROL-Q?
027212 001366      BNE      31$      ;;BRANCH IF NO
027214 012777 000100 151736 MOV      #100,@STKS ;;REENABLE TTY KEYBOARD INTERRUPTS
027222 000002      RTI      ;;RETURN
027224 005237 027002 32$:  INC      $TKCNT     ;;COUNT THIS CHARACTER
027230 021627 000140 CMP      (SP),#140  ;;IS IT UPPER CASE?
027234 002405      BLT      4$      ;;BRANCH IF YES
027236 021627 000175 CMP      (SP),#175  ;;IS IT A SPECIAL CHAR?
027242 003002      BGT      4$      ;;BRANCH IF YES
027244 042716 000040 BIC      #40,(SP)   ;;MAKE IT UPPER CASE
027250 112677 177530 4$:  MOVB     (SP)+,@STKQIN ;;AND PUT IT IN QUEUE
027254 005237 027004 INC      $TKQIN     ;;UPDATE THE POINTER
027260 023727 027004 027011 CMP      $TKQIN,$STKQEND ;;GO OFF THE END?
027266 001003      BNE      5$      ;;BRANCH IF NO
027270 012737 027010 027004 MOV      #$STKQRT,$STKQIN ;;RESET THE POINTER
027276 000002      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.

```

027300 022737 000176 001154 $CKSWR: CMP      #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED
027306 001124      BNE      15$     ;;EXIT IF NOT
027310 105777 151644 TSTB     @STKS      ;;IS A CHAR WAITING?
027314 100121      BPL      15$     ;;IF NOT, EXIT
027316 117746 151640 MOVB     @STKB,-(SP) ;;YES
027322 042716 177600 BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
027326 021627 000007 CMP      (SP),#7    ;;IS IT A CONTROL-G?
027332 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.

```

027334 123727 001150 000001 6$:  CMPB     $AUTOB,#1   ;;ARE WE RUNNING IN AUTO-MODE?
027342 001674      BEQ      2$      ;;BRANCH IF YES
027344 005726      TST      (SP)+      ;;CLEAR CONTROL-G OFF STACK
027346 004737 027012 JSR      PC,$TKINT  ;;FLUSH THE TTY INPUT QUEUE
027352 005077 151602 CLR      @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
027356 112737 000001 001151 MOVB     #1,$INTAG  ;;SET INTERRUPT MODE INDICATOR

027364 104401 030210      TYPE     ,SCNTLG   ;;ECHO THE CONTROL-G (^G)
027370 104401 030215 $GTSWR: TYPE     ,SMSWR   ;;TYPE CURRENT CONTENTS
027374 013746 000176 MOV      SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
027400 104402      TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
027402 104401 030226      TYPE     ,SMNEW   ;;PROMPT FOR NEW SWR
027406 005046 19$:  CLR      -(SP)    ;;CLEAR COUNTER
027410 005046      CLR      -(SP)    ;;THE NEW SWR

```



```
027412 105777 151542      7$:  TSTB  @STKS      ::CHAR THERE?  
027416 100375              BPL  7$           ::IF NOT TRY AGAIN  
  
027420 117746 151536      MOVB  @STKB,-(SP)  ::PICK UP CHAR  
027424 042716 177600      BIC  #^C177,(SP) ::MAKE IT 7-BIT ASCII  
  
027430 021627 000003      CMP   (SP),#3     ::IS IT A CONTROL-C?  
027434 001015              BNE  9$           ::BRANCH IF NOT  
027436 104401 030176      TYPE ,SCNTLC     ::YES, ECHO CONTROL-C (^C)  
027442 062706 000006      ADD  #6,SP       ::CLEAN UP STACK  
027446 123727 001151 000001  CMPB  $INTAG,#1   ::REENABLE TTY KEYBOARD INTERRUPTS?  
027454 001003              BNE  8$           ::BRANCH IF NO  
027456 C12777 000100 151474  MOV  #100,@STKS  ::ALLOW TTY KEYBOARD INTEHRUPTS  
027464 000137 023362      8$:  JMP   SHUT       ::CONTROL-C RESTART  
  
027470 021627 000025      9$:  CMP   (SP),#25  ::IS IT A CONTROL-U?  
027474 001005              BNE  10$          ::BRANCH IF NOT  
027476 104401 030203      TYPE ,SCNTLU     ::YES, ECHO CONTROL-U (^U)  
027502 062706 000006      20$: ADD  #6,SP       ::IGNORE PREVIOUS INPUT  
027506 000737              BR   19$          ::LET'S TRY IT AGAIN  
  
027510 021627 000015      10$: CMP   (SP),#15  ::IS IT A <CR>?  
027514 001022              BNE  16$          ::BRANCH IF NO  
027516 005766 000004      TST  4(SP)       ::YES, IS IT THE FIRST CHAR?  
027522 001403              BEQ  11$          ::BRANCH IF YES  
027524 016677 000002 151422  MOV  2(SP),@SWR  ::SAVE NEW SWR  
027532 062706 000006      11$: ADD  #6,SP       ::CLEAR UP STACK  
027536 104401 001217      14$: TYPE ,$CRLF   ::ECHO <CR> AND <LF>  
027542 123727 001151 000001  CMPB  $INTAG,#1  ::RE-ENABLE TTY KBD INTERRUPTS?  
027550 001003              BNE  15$          ::BRANCH IF NOT  
027552 012777 000100 151400  MOV  #100,@STKS ::RE-ENABLE TTY KBD INTERRUPTS  
027560 000002              15$: RTI          ::RETURN  
027562 004737 026076      16$: JSR  PC,$TYPEC ::ECHO CHAR  
027566 021627 000060      CMP  (SP),#60   ::CHAR < 0?  
027572 002420              BLT  18$          ::BRANCH IF YES  
027574 021627 000067      CMP  (SP),#67   ::CHAR > 7?  
027600 003015              BGT  18$          ::BRANCH IF YES  
027602 042726 000060      BIC  #60,(SP)+  ::STRIP-OFF ASCII  
027606 005766 000002      TST  2(SP)      ::IS THIS THE FIRST CHAR  
027612 001403              BEQ  17$          ::BRANCH IF YES  
027614 006316              ASL  (SP)        ::NO, SHIFT PRESENT  
027616 006316              ASL  (SP)        ::  CHAR OVER TO MAKE  
027620 006316              ASL  (SP)        ::  ROOM FOR NEW ONE.  
027622 005266 000002      17$: INC  2(SP)    ::KEEP COUNT OF CHAR  
027626 056616 177776      BIS  -2(SP),(SP) ::SET IN NEW CHAR  
027632 000667              BR   7$          ::GET THE NEXT ONE  
027634 104401 001216      18$: TYPE ,SQUES  ::TYPE ?<CR><LF>  
027640 000720              BR   20$        ::SIMULATE CONTROL-U  
      .DSABL  LSB
```

```
::*****  
:*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY  
:*CALL:  
:*      RDCHR      ::GET A CHARACTER FROM THE QUEUE
```



```

030066 001011      BNE      3$          ;;BRANCH IF NO
030070 105013      CLRB     (R3)         ;;CLEAR THE CHARACTER
030072 104401 001217 TYPE     ,SCLRF      ;;TYPE A 'CR' & 'LF'
030076 104401 030166 TYPE     ,STTYIN     ;;TYPE THE INPUT STRING
030102 000717      BR       2$          ;;GO PICKUP ANOTHER CHACTER
030104 104401 001216 4$:     TYPE     ,SQUES      ;;TYPE A '?'
030110 000712      BR       1$          ;;CLEAR THE BUFFER AND LOOP
030112 111337 030164 3$:     MOVB    (R3),9$      ;;ECHO THE CHARACTER
030116 104401 030164      TYPE     ,9$
030122 122723 000015      (MPB    #15,(R3)+      ;;CHECK FOR RETURN
030126 001305      BNE      2$          ;;LOOP IF NOT RETURN
030130 105063 177777      CLRB    -1(R3)       ;;CLEAR RETURN (THE 15)
030134 104401 001220      TYPE     ,SLF        ;;TYPE A LINE FEED
030140 005726      TST     (SP)+      ;;CLEAN RUBOUT KEY FROM THE STACK
030142 012603      MOV     (SP)+,R3     ;;RESTORE R3
030144 011646      MOV     (SP),-(SP)    ;;ADJUST THE STACK AND PUT ADDRESS OF THE
030146 016666 000004 000002 MOV     4(SP),2(SP)    ;; FIRST ASCII CHARACTER ON IT
030154 012766 030166 000004 MOV     #STTYIN,4(SP)
030162 000002      RTI
030164 000          9$:     .BYTE    0          ;;RETURN
030165 000          .BYTE    0          ;;STORAGE FOR ASCII CHAR. TO TYPE
030166          .BLKB    8          ;;TERMINATOR
030176 136 103 015 $TTYIN: .BLKB    8          ;;RESERVE 8 BYTES FOR TTY INPUT
030203 136 125 015 $CNTLC: .ASCIZ  /^C/<15><12>  ;;CONTROL 'C'
030210 136 107 015 $CNTLU: .ASCIZ  /^U/<15><12>  ;;CONTROL 'U'
030215 015 012 123 $CNTLG: .ASCIZ  /^G/<15><12>  ;;CONTROL 'G'
030226 040 040 116 $MSWR: .ASCIZ  <15><12>/SWR = /
          $MNEW: .ASCIZ  / NEW = /
          .EVEN
          .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

```

9

```

*****
*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

```

```

030240 011646      $RDOCT: MOV     (SP),-(SP)    ;;PROVIDE SPACE FOR THE
030242 016666 000004 000002 MOV     4(SP),2(SP)    ;; INPUT NUMBER
030250 010046      MOV     R0,-(SP)    ;;PUSH R0 ON STACK
030252 010146      MOV     R1,-(SP)    ;;PUSH R1 ON STACK
030254 010246      MOV     R2,-(SP)    ;;PUSH R2 ON STACK
030256 104412      1$:     RDLIN     ;;READ AN ASCII LINE
030260 012600      MOV     (SP)+,R0    ;;GET ADDRESS OF 1ST CHARACTER
030262 005001      CLR     R1          ;;CLEAR DATA WORD
030264 005002      CLR     R2
030266 112046      2$:     MOVB    (R0)+,-(SP)  ;;PICKUP THIS CHARACTER
030270 001412      BEQ     3$          ;;IF ZERO GET OUT
030272 006301      ASL     R1          ;;*2
030274 006102      ROL     R2
030276 006301      ASL     R1          ;;*4
030300 006102      ROL     R2
030302 006301      ASL     R1          ;;*8
030304 006102      ROL     R2
030306 042716 177770      BIC     #^C7,(SP)    ;;STRIP THE ASCII JUNK
030312 062601      ADD     (SP)+,R1    ;;ADD IN THIS DIGIT

```

030314 000764
 030316 005726
 030320 010166 000012
 030324 010237 030340
 030330 012602
 030332 012601
 030334 012600
 030336 000002
 030340 000000

10

```

3$: BR 2$ ;;LOOP
    TST (SP)+ ;;CLEAN TERMINATOR FROM STACK
    MOV R1,12(SP) ;;SAVE THE RESULT
    MOV R2,$HIOCT
    MOV (SP)+,R2 ;;POP STACK INTO R2
    MOV (SP)+,R1 ;;POP STACK INTO R1
    MOV (SP)+,R0 ;;POP STACK INTO R0
    RTI ;;RETURN
$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.
    
```

030342 016646 000002
 030346 042716 000020
 030352 012746 030360
 030356 000002
 030360 010046
 030362 016600 000002
 030366 005740
 030370 111000
 030372 006300
 030374 016000 030414
 030400 000200

```

$TRAP: MOV 2(SP),-(SP) ;;ASSUME THE STATUS OF
        BIC #20,(SP) ;; THE CALLER--DO NOT ALLOW
        MOV #1$,-(SP) ;; T-BIT TRAPS
        RTI ;;SET THE NEW STATUS
1$: MOV R0,-(SP) ;;SAVE R0
    MOV 2(SP),R0 ;;GET TRAP ADDRESS
    TST -(R0) ;;BACKUP BY 2
    MOVB (R0),R0 ;;GET RIGHT BYTE OF TRAP
    ASL R0 ;;POSITION FOR INDEXING
    MOV $TRPAD(R0),R0 ;;INDEX TO TABLE
    RTS R0 ;;GO TO ROUTINE
    
```

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

030402 011646
 030404 016666 000004 000002
 030412 000002

```

$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
        MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
        RTI ;;RESTORE THE PSW
    
```

.SBTTL TRAP TABLE

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

```

: ROUTINE
:-----
$TRPAD: .WORD $TRAP2
        $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
        $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
        $TYPBN ;;CALL=TYPBN TRAP+6(104406) TYPE BINARY (ASCII) NUMBER

030414 030402 $GTSWR ;;CALL=GTSWR TRAP+7(104407) GET SOFT-SWR SETTING
030416 025664
030420 025462
030422 025436
030424 025476
030426 025212
030430 025136

030432 027370
030434 027300 $CKSWR ;;CALL=CKSWR TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
030436 027642 $RDCHR ;;CALL=RDCHR TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
030440 027732 $RDLIN ;;CALL=RDLIN TRAP+12(104412) TTY TYPEIN STRING ROUTINE
030442 030240 $RDOCT ;;CALL=RDOCT TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
    
```

030444 025042
030446 025100
11

\$SAVREG ;;CALL=SAVREG TRAP+14(104414) SAVE R0-R5 ROUTINE
\$RESREG ;;CALL=RESREG TRAP+15(104415) RESTORE R0-R5 ROUTINE
.SBTTL POWER DOWN AND UP ROUTINES

:POWER DOWN ROUTINE

030450 012737 030610 000024 \$PWRDN: MOV #SILLUP,@#PWRVEC ;;SET FOR FAST UP
030456 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
030464 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
030466 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
030470 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
030472 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
030474 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
030476 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
030500 017746 150450 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
030504 010637 030614 MOV SP,\$SAVR6 ;;SAVE SP
030510 012737 030522 000024 MOV #SPWRUP,@#PWRVEC ;;SET UP VECTOR
030516 000000 HALT
030520 000776 BR -2 ;;HANG UP

:POWER UP ROUTINE

030522 012737 030610 000024 \$PWRUP: MOV #SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
030530 013706 030614 MOV \$SAVR6,SP ;;GET SP
030534 005037 030614 CLR \$SAVR6 ;;WAIT LOOP FOR THE TTY
030540 005237 030614 1\$: INC \$SAVR6 ;;WAIT FOR THE INC
030544 001375 BNE 1\$;;OF WORD
030546 012677 150402 MOV (SP)+,@SWR ;;POP STACK INTO @SWR
030552 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
030554 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
030556 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
030560 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
030562 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
030564 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
030566 012737 030450 000024 MOV #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
030574 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
030602 104401 TYPE \$POWER ;;REPORT THE POWER FAILURE
030604 030616 \$PWRMG: .WORD \$POWER ;;POWER FAIL MESSAGE POINTER
030606 000002 RTI
030610 000000 \$ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
030612 000776 BR -2 ;; BEFORE THE POWER DOWN WAS COMPLETE
030614 000000 \$SAVR6: 0 ;;PUT THE SP HERE
030616 015 012 120 \$POWER: .ASCIZ <15><12>'POWER'
.EVEN

.SBTTL APT COMMUNICATIONS ROUTINE

12
030626 112737 000001 031072 \$ATY1: MOVB #1,\$FFLG ;;TO REPORT FATAL ERROR
030634 112737 000001 031070 \$ATY3: MOVB #1,\$MFLG ;;TO TYPE A MESSAGE
030642 000403 BR \$ATYC
030644 112737 000001 031072 \$ATY4: MOVB #1,\$FFLG ;;TO ONLY REPORT FATAL ERROR
030652 \$ATYC:
030652 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
030654 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
030656 105737 031070 TSTB \$MFLG ;;SHOULD TYPE A MESSAGE?
030662 001450 BEQ \$\$;;IF NOT: BR
030664 122737 000001 001242 CMPB #APTENV,\$ENV ;;OPERATING UNDER APT?

030672	001031			BNE	3\$::IF NOT: BR
030674	132737	000100	001243	BITB	#APTSPOOL,\$ENV	::SHOULD SPOOL MESSAGES?
030702	001425			BEQ	3\$::IF NOT: BR
030704	017600	000004		MOV	@4(SP),R0	::GET MESSAGE ADDR.
030710	062766	000002	000004	ADD	#2,4(SP)	::BUMP RETURN ADDR.
030716	005737	001222		1\$: TST	\$MSGTYPE	::SEE IF DONE W/ LAST XMISSION?
030722	001375			BNE	1\$::IF NOT: WAIT
030724	010037	001236		MOV	R0,\$MSGAD	::PUT ADDR IN MAILBOX
030730	105720			2\$: TSTB	(R0)+	::FIND END OF MESSAGE
030732	001376			BNE	2\$	
030734	163700	001236		SUB	\$MSGAD,R0	::SUB START OF MESSAGE
030740	006200			ASR	R0	::GET MESSAGE LNTH IN WORDS
030742	010037	001240		MOV	R0,\$MSGGLT	::PUT LENGTH IN MAILBOX
030746	012737	000004	001222	MOV	#4,\$MSGTYPE	::TELL APT TO TAKE MSG.
030754	000413			BR	5\$	
030756	017637	000004	031002	3\$: MOV	@4(SP),4\$::PUT MSG ADDR IN JSR LINKAGE
030764	062766	000002	000004	ADD	#2,4(SP)	::BUMP RETURN ADDRESS
030772	013746	177776		MOV	177776,-(SP)	::PUSH 177776 ON STACK
030776	004737	025664		JSR	PC,\$TYPE	::CALL TYPE MACRO
031002	000000			4\$: .WORD	0	
031004				5\$:		
031004	105737	031072		10\$: TSTB	\$FFLG	::SHOULD REPORT FATAL ERROR?
031010	001416			BEQ	12\$::IF NOT: BR
031012	005737	001242		TST	\$ENV	::RUNNING UNDER APT?
031016	001413			BEQ	12\$::IF NOT: BR
031020	005737	001222		11\$: TST	\$MSGTYPE	::FINISHED LAST MESSAGE?
031024	001375			BNE	11\$::IF NOT: WAIT
031026	017637	000004	001224	MOV	@4(SP),\$FATAL	::GET ERROR #
031034	062766	000002	000004	ADD	#2,4(SP)	::BUMP RETURN ADDR.
031042	005237	001222		INC	\$MSGTYPE	::TELL APT TO TAKE ERROR
031046	105037	031072		12\$: CLRB	\$FFLG	::CLEAR FATAL FLAG
031052	105037	031071		CLRB	\$LFLG	::CLEAR LOG FLAG
031056	105037	031070		CLRB	\$MFLG	::CLEAR MESSAGE FLAG
031062	012601			MOV	(SP)+,R1	::POP STACK INTO R1
031064	012600			MOV	(SP)+,R0	::POP STACK INTO R0
031066	000207			RTS	PC	::RETURN
031070	000			\$MFLG: .BYTE	0	::MESSG. FLAG
031071	000			\$LFLG: .BYTE	0	::LOG FLAG
031072	000			\$FFLG: .BYTE	0	::FATAL FLAG
				.EVEN		
	000200			APTSIZE =	200	
	000001			APTENV =	001	
	000100			APTSPOOL =	100	
	000040			APTCSUP =	040	

```

1          .SBTTL  CONSOLE MESSAGES
2
3 031074    075    000    EQUALS: .ASCIZ  @=@
4 031076    101    114    114  ALL:   .ASCIZ  @ALL@<CRLF>
5 031103    040    077    040  QUES:  .ASCIZ  @ ? @
6 031107    054    040    000  COMMA: .ASCIZ  @, @
7 031112    200    124    131  MSHELP: .ASCIZ  <CRLF>@TYPE HELP TEXT (Y/N) ? @
8 031143
9 031143    200    103    110  UBUSQST: .ASCIZ  <CRLF>@CHANGE ADDRESSES (Y/N) ? @
10 031176   200    125    123  CNSL00: .ASCIZ  <CRLF>@USE SAME DEVICES (Y/N) ? @
11 031231   200    102    125  CNSL01: .ASCIZ  <CRLF>@BUS ADDRESS @
12 031247   040    114    111  CNSL02: .ASCIZ  @ LIMITS - LO= 160000, HI= 17XXXX@<CRLF>
13 031311   126    105    103  CNSL03: .ASCIZ  @VECTOR ADDRESS @
14 031331   040    114    111  CNSL04: .ASCIZ  @ LIMITS - LO= 0, HI= 1000@<CRLF><LF>
15 031365   102    122    040  CNSL05: .ASCIZ  @BR LEVEL @
16 031377   040    114    111  CNSL06: .ASCIZ  @ LIMITS - LO= 0, HI= 7@<CRLF><LF>
17 031430   200
18 031431   200    124    131  CNSL07: .ASCII  <CRLF>
19 031516   200    101    116  .ASCII  <CRLF>@TYPE 'A' TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S)@
20 031573   200
21 031574   040    077    111  .ASCII  <CRLF>@AND TERMINATE INPUT WITH A CARRIAGE RETURN.@
22 031615   200    104    122  CNSL08: .ASCII  <CRLF>
23 031631   104    122    111  CNSL09: .ASCIZ  @ ?ILLEGAL INPUT@<CRLF>
24 031637   040    111    123  MSDRVS: .ASCIZ  <CRLF>/DRIVE(S): /
25 031657   040    116    117  MSGDRV: .ASCIZ  /DRIVE/
26 031674   040    116    117  LODEV:  .ASCIZ  / IS LOAD DEVICE/
27
28
29          .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

```
.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 THE 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
```

031714

031714 020000

58	031716	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (2)
59	031720	132000	.WORD	ATA!OPI!IVC!IAE	:SEEK
60	031722	130000	.WORD	ATA!OPI!IVC	:RECALIBRATE
61	031724	020000	.WORD	OPI	:DRIVE CLEAR
62	031726	030000	.WORD	OPI!IVC	:RELEASE
63	031730	130000	.WORD	OPI!ATA!IVC	:OFFSET
64	031732	130000	.WORD	OPI!ATA!IVC	:RETURN TO CENTERLINE
65	031734	020000	.WORD	OPI	:READ IN PRESET
66	031736	020000	.WORD	OPI	:PACK ACKNOWLEDGE
67	031740	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (24)
68	031742	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (26)
69	031744	132000	.WORD	ATA!OPI!IVC!IAE	:SEARCH
70	031746	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (32)
71	031750	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (34)
72	031752	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (36)
73	031754	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (40)
74	031756	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (42)
75	031760	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (44)
76	031762	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (46)
77	031764	073300	.WORD	WCE!OPI!IVC!IAE!AOE!HCE!ECH	:WRITE CHECK DATA
78	031766	073300	.WORD	WCE!OPI!IVC!IAE!AOE!HCE!ECH	:WRITE CHECK HEADER AND DATA
79	031770	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (54)
80	031772	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (56)
81	031774	037200	.WORD	OPI!IVC!WLE!IAE!AOE!HCE	:WRITE DATA
82	031776	037000	.WORD	OPI!IVC!WLE!IAE!AOE	:WRITE HEADER AND DATA
83	032000	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (64)
84	032002	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (66)
85	032004	033300	.WORD	OPI!IVC!IAE!AOE!HCE!ECH	:READ DATA
86	032006	033300	.WORD	OPI!IVC!IAE!AOE!HCE!ECH	:READ HEADER AND DATA
87	032010	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (74)
88	032012	130001	.WORD	OPI!ATA!ILF!IVC	:ILLEGAL FUNCTION (76)
89					

1			.SBTTL	ATTENTION (ATA) TABLE
2				
3	032014	001	ATNTBL:	.BYTE 1.
4	032015	002		.BYTE 2.
5	032016	004		.BYTE 4.
6	032017	010		.BYTE 8.
7	032020	020		.BYTE 16.
8	032021	040		.BYTE 32.
9	032022	100		.BYTE 64.
10	032023	200		.BYTE 128.
11			.EVEN	
12				

Line	Code	Address	Offset	Value	Label	Text
1					.SBTTL	ERROR MESSAGE TABLE
2						
3	032024	046514	040126	000000	EMT1:	.WORD FMS300,EMS1,0
4	032032	046532	046555	046602	EMT2:	.WORD EMS301,EMS302,EMS303,EMS1,EMS304
5	032044	051772	051176	051336		.WORD EMS511,EMS500,EMS501,EMS502,EMS503,0
6	032060	046532	046622	046555	EMT3:	.WORD EMS301,EMS306,EMS302
7	032066	051772	051513	051336		.WORD EMS511,EMS505,EMS501,EMS502,0
8	032100	046514	046555	046636	EMT4:	.WORD EMS300,EMS302,EMS307,EMS2
9	032110	051772	051363	051336		.WORD EMS511,EMS502,EMS501,EMS503,0
10	032122	046532	046677	046714	EMT5:	.WORD EMS301,EMS310,EMS311
11	032130	051772	051363	051336		.WORD EMS511,EMS502,EMS501,EMS503,EMS504
12	032142	046760	000000			.WORD EMS312,0
13	032146	046532	046622	046714	EMT6:	.WORD EMS301,EMS306,EMS311
14	032154	051772	051363	051336		.WORD EMS511,EMS502,EMS501,EMS503,EMS504,0
15	032170	046532	047016	046555	EMT7:	.WORD EMS301,EMS313,EMS302
16	032176	051772	051336	051363		.WORD EMS511,EMS501,EMS502,EMS504,EMS503,0
17	032212	047124	047145	047047	EMT10:	.WORD EMS316,EMS317,EMS314
18	032220	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
19	032230	047124	047145	047076	EMT11:	.WORD EMS316,EMS317,EMS315
20	032236	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
21	032246	047124	047165	047047	EMT12:	.WORD EMS316,EMS320,EMS314
22	032254	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
23	032264	047124	047165	047076	EMT13:	.WORD EMS316,EMS320,EMS315
24	032272	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
25	032302	047124	047205	047047	EMT14:	.WORD EMS316,EMS321,EMS314
26	032310	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
27	032320	047124	047205	047076	EMT15:	.WORD EMS316,EMS321,EMS315
28	032326	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
29	032336	047124	047225	047047	EMT16:	.WORD EMS316,EMS322,EMS314
30	032344	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
31	032354	047124	047225	047076	EMT17:	.WORD EMS316,EMS322,EMS315
32	032362	051772	051336	051363		.WORD EMS511,EMS501,EMS502,0
33	032372	044506	047444	047522	EMT20:	.WORD EMS71,EMS335,EMS340,EMS72,EMS377,EMS372
34	032406	051772	051433	000000		.WORD EMS511,EMS503,0
35	032414	044506	047467	047522	EMT21:	.WORD EMS71,EMS336,EMS340,EMS72,EMS400,EMS372
36	032430	051772	051433	000000		.WORD EMS511,EMS503,0
37	032436	044642	047747	044712	EMT22:	.WORD EMS73,EMS352,EMS74,EMS402,EMS70,EMS406
38	032452	051772	051433	051460		.WORD EMS511,EMS503,EMS504,0
39	032462	044642	047747	044765	EMT23:	.WORD EMS73,EMS352,EMS75,EMS402,EMS77,EMS406
40	032476	051772	051433	051460		.WORD EMS511,EMS503,EMS504,0
41	032506	044642	050064	044765	EMT24:	.WORD EMS73,EMS356,EMS75,EMS402,EMS77,EMS377
42	032522	051772	051433	051460		.WORD EMS511,EMS503,EMS504,0
43	032532	044642	047444	047522	EMT25:	.WORD EMS73,EMS335,EMS340,EMS76,EMS411
44	032544	051772	051433	000000		.WORD EMS511,EMS503,0
45	032552	046532	046622	046104	EMT26:	.WORD EMS301,EMS306,EMS252,EMS253,EMS327,EMS254
46	032566	051772	051433	051336		.WORD EMS511,EMS503,EMS501,EMS502
47	032576	047333	047047	000000		.WORD EMS330,EMS314,0
48	032604	047506	044642	050741	EMT27:	.WORD EMS337,EMS73,EMS410,EMS76,EMS411
49	032616	051772	051433	000000		.WORD EMS511,EMS503,0
50	032624	047506	045145	047533	EMT30:	.WORD EMS337,EMS100,EMS341,EMS101
51	032634	051772	051433	051460		.WORD EMS511,EMS503,EMS504,0
52	032644	046514	046104		EMT31:	.WORD EMS300,EMS252
53	032650	051772	051336	051433		.WORD EMS511,EMS501,EMS503,0
54	032660	045145	050361		EMT32:	.WORD EMS100,EMS370
55	032664	051772	051460	000000		.WORD EMS511,EMS504,0
56	032672	045145	050765		EMT33:	.WORD EMS100,EMS412
57	032676	051772	051460	000000		.WORD EMS511,EMS504,0

58	032704	045261	050765		EMT34:	.WORD	EMS102,EMS412
59	032710	051772	051433	000000		.WORD	EMS511,EMS503,0
60	032716	045261	047467		EMT35:	.WORD	EMS102,EMS336
61	032722	051772	051433	000000		.WORD	EMS511,EMS503,0
62	032730	045145	047444	047522	EMT36:	.WORD	EMS100,EMS335,EMS340,EMS102,EMS334
63	032742	051772	051460	000000		.WORD	EMS511,EMS504,0
64	032750	045145	047444	047522	EMT37:	.WORD	EMS100,EMS335,EMS340,EMS102,EMS377,EMS365
65	032764	051000	045367	051032		.WORD	EMS413,EMS104,EMS415
66	032772	051772	051460	000000		.WORD	EMS511,EMS504,0
67	033000	045145	047467	047522	EMT40:	.WORD	EMS100,EMS336,EMS340,EMS416,EMS104,EMS415
68	033014	051772	051460	000000		.WORD	EMS511,EMS504,0
69	033022	045145	047467	047522	EMT41:	.WORD	EMS100,EMS336,EMS340,EMS73,EMS415,EMS402
70	033036	045261	050555			.WORD	EMS102,EMS400
71	033042	051772	051460	000000		.WORD	EMS511,EMS504,0
72	033050	045327	050572	047325	EMT42:	.WORD	EMS103,EMS401,EMS327,EMS370
73	033060	051772	051433	000000		.WORD	EMS511,EMS503,0
74	033066	051013	045367	051062	EMT43:	.WORD	EMS414,EMS104,EMS417
75	033074	051772	051460	000000		.WORD	EMS511,EMS504,0
76	033102	045414	050572	047325	EMT44:	.WORD	EMS105,EMS401,EMS327,EMS370
77	033112	051772	051433	000000		.WORD	EMS511,EMS503,0
78	033120	046514	046137		EMT45:	.WORD	EMS300,EMS253
79	033124	051772	051336	051433		.WORD	EMS511,EMS501,EMS503,0
80	033134	045455	050572	051062	EMT46:	.WORD	EMS106,EMS401,EMS417
81	033142	051772	051433	000000		.WORD	EMS511,EMS503,0
82	033150	051000	045522	051100	EMT47:	.WORD	EMS413,EMS107,EMS420,EMS417
83	033160	051772	051460	000000		.WORD	EMS511,EMS504,0
84	033166	045532	051000	050676	EMT50:	.WORD	EMS110,EMS413,EMS405,EMS107
85	033176	051772	051433	000000		.WORD	EMS511,EMS503,0
86	033204	047574	045571	051113	EMT51:	.WORD	EMS343,EMS111,EMS421
87	033212	051772	051460	000000		.WORD	EMS511,EMS504,0
88	033220	051013	051051	045522	EMT52:	.WORD	EMS414,EMS416,EMS107,EMS421
89	033230	051772	051460	000000		.WORD	EMS511,EMS504,0
90	033236	047352	040313	046275	EMT53:	.WORD	EMS331,EMS4,EMS256
91	033244	051772	051336	000000		.WORD	EMS511,EMS501,0
92	033252	045607	051000	050676	EMT54:	.WORD	EMS112,EMS413,EMS405,EMS607
93	033262	051772	051433	000000		.WORD	EMS511,EMS503,0
94	033270	045645	050572	047325	EMT55:	.WORD	EMS113,EMS401,EMS327,EMS370
95	033300	051772	051433	000000		.WORD	EMS511,EMS503,0
96	033306	045704	050572	051062	EMT56:	.WORD	EMS114,EMS401,EMS417
97	033314	051772	051433	000000		.WORD	EMS511,EMS503,0
98	033322	046325	047402		EMT57:	.WORD	EMS257,EMS332
99	033326	051772	051566	051336		.WORD	EMS511,EMS506,EMS501,0
100	033336	040344	047420		EMT60:	.WORD	EMS5,EMS333
101	033342	051772	051627	051336		.WORD	EMS511,EMS507,EMS501,0
102	033352	051000	045752	051100	EMT61:	.WORD	EMS413,EMS115,EMS420,EMS417
103	033362	051772	051460	000000		.WORD	EMS511,EMS504,0
104	033370	047665	045145	047533	EMT62:	.WORD	EMS346,EMS100,EMS341,EMS101
105	033400	051772	051460	000000		.WORD	EMS511,EMS504,0
106	033406	045645	050376		EMT63:	.WORD	EMS113,EMS371
107	033412	051772	051433	000000		.WORD	EMS511,EMS503,0
108	033420	051000	045767	047763	EMT64:	.WORD	EMS413,EMS116,EMS353
109	033426	051772	051460	000000		.WORD	EMS511,EMS504,0
110	033434	051130	047763		EMT65:	.WORD	EMS422,EMS353
111	033440	051772	051460	000000		.WORD	EMS511,EMS504,0
112	033446	040665	047467	047542	EMT66:	.WORD	EMS12,EMS336,EMS342
113	033454	051772	051336	000000		.WORD	EMS511,EMS501,0
114	033462	046514	040414		EMT67:	.WORD	EMS300,EMS6

115	033466	051772	051336		.WORD	EMS511,EMS501
116	033472	040460	047420	000000	.WORD	EMS7,EMS333,0
117	033500	046514	040414	047325	EMT70: .WORD	EMS300,EMS6,EMS327,EMS7
118	033510	051772	051336	051460	.WORD	EMS511,EMS501,EMS504,0
119	033520	040414	047444	047522	EMT71: .WORD	EMS6,EMS335,EMS340,EMS10,EMS333,EMS342
120	033530	051772	051336	051363	.WORD	EMS511,EMS501,EMS502,0
121	033544	040414	047467	047522	EMT72: .WORD	EMS6,EMS336,EMS340,EMS10,EMS334,EMS342
122	033560	051772	051336	051363	.WORD	EMS511,EMS501,EMS502,0
123	033570	046532	046361	046602	EMT73: .WORD	EMS301,EMS260,EMS303,EMS11
124	033600	051772	051336	051363	.WORD	EMS511,EMS501,EMS502,0
125	033610	047574	047610	047542	EMT74: .WORD	EMS343,EMS344,EMS342,0
126	033620	046514	040743		EMT75: .WORD	EMS300,EMS13
127	033624	051772	051433	000000	.WORD	EMS511,EMS503,0
128	033632	047665	040743	047637	EMT76: .WORD	EMS346,EMS13,EMS345
129	033640	051772	051433	000000	.WORD	EMS511,EMS503,0
130	033646	047506	040743	047637	EMT77: .WORD	EMS337,EMS13,EMS345
131	033654	051772	051433	000000	.WORD	EMS511,EMS503,0
132	033662	046532	047016	046172	EMT100: .WORD	EMS301,EMS313,EMS254,EMS347,EMS13
133	033674	051772	051433	000000	.WORD	EMS511,EMS503,0
134	033702	047665	041012	047533	EMT101: .WORD	EMS346,EMS14,EMS341,EMS15
135	033712	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
136	033722	047506	044440		EMT102: .WORD	EMS337,EMS70
137	033726	051772	051433	000000	.WORD	EMS511,EMS503,0
138	033734	046532	047016	046172	EMT103: .WORD	EMS301,EMS313,EMS254,EMS347,EMS15
139	033746	051772	051433		.WORD	EMS511,EMS503
140	033752	041012	047402	000000	.WORD	EMS14,EMS332,0
141	033760	047665	041216	047533	EMT104: .WORD	EMS346,EMS17,EMS341,EMS16
142	033770	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
143	034000	047506	041216	047533	EMT105: .WORD	EMS337,EMS17,EMS341,EMS16
144	034010	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
145	034020	046532	047016	046172	EMT106: .WORD	EMS301,EMS313,EMS254,EMS347,EMS16
146	034032	051772	051433		.WORD	EMS511,EMS503
147	034036	041216	047402	000000	.WORD	EMS17,EMS332,0
148	034044	047665	041257	047533	EMT107: .WORD	EMS346,EMS20,EMS341,EMS21
149	034054	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
150	034064	041257	047731	041323	EMT110: .WORD	EMS20,EMS351,EMS21,EMS350,EMS22,EMS315
151	034100	051772	051336	000000	.WORD	EMS511,EMS501,0
152	034106	041257	047434	047724	EMT111: .WORD	EMS20,EMS334,EMS350,EMS22,EMS333
153	034120	051772	051336	000000	.WORD	EMS511,EMS501,0
154	034126	047506	041257	047533	EMT112: .WORD	EMS337,EMS20,EMS341,EMS21
155	034136	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
156	034146	047506	041257	047533	EMT113: .WORD	EMS337,EMS20,EMS341,EMS21,EMS350,EMS22,EMS334
157	034164	051772	051336	000000	.WORD	EMS511,EMS501,0
158	034172	041257	047747	041323	EMT114: .WORD	EMS20,EMS352,EMS21,EMS350,EMS22,EMS333
159	034206	051772	051336	000000	.WORD	EMS511,EMS501,0
160	034214	046532	047016	046172	EMT115: .WORD	EMS301,EMS313,EMS254,EMS347,EMS21
161	034226	051772	051433		.WORD	EMS511,EMS503
162	034232	041257	047402	000000	.WORD	EMS20,EMS332,0
163	034240	047665	041447	047533	EMT116: .WORD	EMS346,EMS23,EMS341,EMS24
164	034250	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
165	034260	047506	041447	047533	EMT117: .WORD	EMS337,EMS23,EMS341,EMS24
166	034270	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
167	034300	046532	047016	046172	EMT120: .WORD	EMS301,EMS313,EMS254,EMS347,EMS24
168	034312	051772	051433		.WORD	EMS511,EMS503
169	034316	041447	047402	000000	.WORD	EMS23,EMS332,0
170	034324	047665	041604	047533	EMT121: .WORD	EMS346,EMS25,EMS341,EMS26
171	034334	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0

172	034344	047506	041604	047533	EMT122:	.WORD	EMS337,EMS25,EMS341,EMS26
173	034354	051772	051433	051336		.WORD	EMS511,EMS503,EMS501,0
174	034364	046532	047016	046172	EMT123:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS26
175	034376	051772	051433			.WORD	EMS511,EMS503
176	034402	041604	047402	000000		.WORD	EMS25,EMS332,0
177	034410	046514	041741	046636	EMT124:	.WORD	EMS300,EMS27,EMS307,EMS2
178	034420	051772	051433	000000		.WORD	EMS511,EMS503,0
179	034426	047352	041741	047763	EMT125:	.WORD	EMS331,EMS27,EMS353
180	034434	051772	051433			.WORD	EMS511,EMS503
181	034440	042005	047076	000000		.WORD	EMS30,EMS315,0
182	034446	047506	041741	047533	EMT126:	.WORD	EMS337,EMS27,EMS341,EMS30
183	034456	051772	051433	000000		.WORD	EMS511,EMS503,0
184	034464	046532	047016	046172	EMT127:	.WORD	EMS301,EMS313,EMS254,EMS347,EMS30
185	034476	051772	051433			.WORD	EMS511,EMS503
186	034502	041741	047402	000000		.WORD	EMS27,EMS332,0
187	034510	042065	050007	046002	EMT130:	.WORD	EMS31,EMS354,EMS250
188	034516	051772	051460	051433		.WORD	EMS511,EMS504,EMS503,0
189	034526	042136	050007	046002	EMT131:	.WORD	EMS32,EMS354,EMS250
190	034534	051772	051460	051433		.WORD	EMS511,EMS504,EMS503,0
191	034544	050042	042216	046002	EMT132:	.WORD	EMS355,EMS33,EMS250,EMS341,EMS30
192	034556	051772	051460	000000		.WORD	EMS511,EMS504,0
193	034564	050042	042246	046002	EMT133:	.WORD	EMS355,EMS34,EMS250,EMS341,EMS30
194	034576	051772	051460	000000		.WORD	EMS511,EMS504,0
195	034604	047352	040313	046233	EMT134:	.WORD	EMS331,EMS4,EMS255
196	034612	051772	051460	000000		.WORD	EMS511,EMS504,0
197	034620	042275	050104	050126	EMT135:	.WORD	EMS35,EMS357,EMS360,EMS15
198	034630	051772	051336	000000		.WORD	EMS511,EMS501,0
199	034636	046412	050202		EMT136:	.WORD	EMS261,EMS362
200	034642	051772	051433	000000		.WORD	EMS511,EMS503,0
201	034650	046514	042337	046636	EMT137:	.WORD	EMS300,EMS36,EMS307,EMS2
202	034660	051772	051336	000000		.WORD	EMS511,EMS501,0
203	034666	050042	042367	046233	EMT140:	.WORD	EMS355,EMS37,EMS255,EMS341,EMS30
204	034700	051772	051460	000000		.WORD	EMS511,EMS504,0
205	034706	047665	042421	047637	EMT141:	.WORD	EMS346,EMS40,EMS345
206	034714	051772	051460	000000		.WORD	EMS511,EMS504,0
207	034722	047506	042421	047533	EMT142:	.WORD	EMS337,EMS40,EMS341,EMS30
208	034732	051772	051460	000000		.WORD	EMS511,EMS504,0
209	034740	050223	046677	042477	EMT143:	.WORD	EMS363,EMS310,EMS41
210	034746	051772	051336	000000		.WORD	EMS511,EMS501,0
211	034754	047506	042477	047533	EMT144:	.WORD	EMS337,EMS41,EMS341,EMS252,EMS327,EMS253
212	034770	051772	051336	000000		.WORD	EMS511,EMS501,0
213	034776	042477	050007	050240	EMT145:	.WORD	EMS41,EMS354,EMS364,EMS252,EMS365,EMS253
214	035012	051772	051336	000000		.WORD	EMS511,EMS501,0
215	035020	046532	046622	042337	EMT146:	.WORD	EMS301,EMS306,EMS36
216	035026	051772	051336	051433		.WORD	EMS511,EMS501,EMS503,0
217	035036	050266	042545		EMT147:	.WORD	EMS366,EMS42
218	035042	051772	051433	000000		.WORD	EMS511,EMS503,0
219	035050	050311	047763	050261	EMT150:	.WORD	EMS367,EMS353,EMS365,EMS42,EMS354,EMS3
220	035064	051772	051433	000000		.WORD	EMS511,EMS503,0
221	035072	047506	042337		EMT151:	.WORD	EMS337,EMS36
222	035076	051772	051336	000000		.WORD	EMS511,EMS501,0
223	035104	042627	050007	042337	EMT152:	.WORD	EMS43,EMS354,EMS36
224	035112	051772	051336	000000		.WORD	EMS511,EMS501,0
225	035120	050311	047763	050261	EMT153:	.WORD	EMS367,EMS353,EMS365,EMS36,EMS370
226	035132	051772	051433	000000		.WORD	EMS511,EMS503,0
227	035140	050311	047763	050261	EMT154:	.WORD	EMS367,EMS353,EMS365,EMS36,EMS371
228	035152	051772	051433	000000		.WORD	EMS511,EMS503,0

229	035160	046514	042700	046636	EMT155: .WORD	EMS300,EMS44,EMS307,EMS2
230	035170	051772	051433	000000	.WORD	EMS511,EMS503,0
231	035176	050311	047763	050261	EMT156: .WORD	EMS367,EMS353,EMS365,EMS44,EMS354,EMS3
232	035212	051772	051433	000000	.WORD	EMS511,EMS503,0
233	035220	046514	042741	046636	EMT157: .WORD	EMS300,EMS45,EMS307,EMS2
234	035230	051772	051433	000000	.WORD	EMS511,EMS503,0
235	035236	050311	047763	050261	EMT160: .WORD	EMS367,EMS353,EMS365,EMS45,EMS354,EMS3
236	035252	051772	051433	051336	.WORD	EMS511,EMS503,EMS501
237	035260	042275	047420	000000	.WORD	EMS35,EMS333,0
238	035266	046514	043016	046636	EMT161: .WORD	EMS300,EMS46,EMS307,EMS2
239	035276	051772	051433	000000	.WORD	EMS511,EMS503,0
240	035304	047506	043016	047763	EMT162: .WORD	EMS337,EMS46,EMS353
241	035312	051772	051433	051336	.WORD	EMS511,EMS503,EMS501,0
242	035322	042275	047444	047506	EMT163: .WORD	EMS35,EMS335,EMS337,EMS41,EMS334,EMS372
243	035336	051772	051336	000000	.WORD	EMS511,EMS501,0
244	035344	043100	047444	047506	EMT164: .WORD	EMS47,EMS335,EMS337,EMS41,EMS335,EMS372
245	035360	051772	051336	000000	.WORD	EMS511,EMS501,0
246	035366	047506	042275	047325	EMT165: .WORD	EMS337,EMS35,EMS327,EMS47
247	035376	051772	051336		.WORD	EMS511,EMS501
248	035402	042477	047420	050425	.WORD	EMS41,EMS333,EMS372,0
249	035412	046514	043100	046636	EMT166: .WORD	EMS300,EMS47,EMS307,EMS2
250	035422	051772	051336	051433	.WORD	EMS511,EMS501,EMS503,0
251	035432	043140	047444	047522	EMT167: .WORD	EMS50,EMS335,EMS340,EMS36,EMS333
252	035444	051772	051336	051433	.WORD	EMS511,EMS501,EMS503,0
253	035454	047506	042275		EMT170: .WORD	EMS337,EMS35
254	035460	051772	051336	000000	.WORD	EMS511,EMS501,0
255	035466	043140	042246	040246	EMT171: .WORD	EMS50,EMS34,EMS3
256	035474	051772	051336	000000	.WORD	EMS511,EMS501,0
257	035502	046532	046622	043207	EMT172: .WORD	EMS301,EMS306,EMS51
258	035510	051772	051336	051460	.WORD	EMS511,EMS501,EMS504,0
259	035520	050311	047763	050261	EMT173: .WORD	EMS367,EMS353,EMS365,EMS47,EMS354,EMS3
260	035534	051772	051336	000000	.WORD	EMS511,EMS501,0
261	035542	046514	046002	047325	EMT174: .WORD	EMS300,EMS250,EMS327,EMS255,EMS327,EMS256
262	035556	047533	052071		.WORD	EMS341,EMS600
263	035562	051772	051336	000000	.WORD	EMS511,EMS501,0
264	035570	046514	046275	047533	EMT175: .WORD	EMS300,EMS256,EMS341,EMS600
265	035600	051772	051336	000000	.WORD	EMS511,EMS501,0
266	035606	046514	046002	047533	EMT176: .WORD	EMS300,EMS250,EMS341,EMS600
267	035616	051772	051460	000000	.WORD	EMS511,EMS504,0
268	035624	046514	046233	047533	EMT177: .WORD	EMS300,EMS255,EMS341,EMS600
269	035634	051772	051460	000000	.WORD	EMS511,EMS504,0
270	035642	047506	043256	047533	EMT200: .WORD	EMS337,EMS52,EMS341,EMS601
271	035652	051772	051336	000000	.WORD	EMS511,EMS501,0
272	035660	047665	043256	047533	EMT201: .WORD	EMS346,EMS52,EMS341,EMS602
273	035670	051772	051336	000000	.WORD	EMS511,EMS501,0
274	035676	047665	043207	047533	EMT202: .WORD	EMS346,EMS51,EMS341,EMS602
275	035706	051772	051336	000000	.WORD	EMS511,EMS501,0
276	035714	047665	043256	047637	EMT203: .WORD	EMS346,EMS52,EMS345,EMS373,EMS255
277	035726	051772	051460	051336	.WORD	EMS511,EMS504,EMS501,0
278	035736	047665	043256	047533	EMT204: .WORD	EMS346,EMS52,EMS341,EMS27
279	035746	051772	051460	051336	.WORD	EMS511,EMS504,EMS501,0
280	035756	043317	050007	040246	EMT205: .WORD	EMS53,EMS354,EMS3
281	035764	051772	051433	051363	.WORD	EMS511,EMS503,EMS502,EMS510,0
282	035776	047506	043360	050463	EMT206: .WORD	EMS337,EMS54,EMS374,EMS250,EMS327,EMS255
283	036012	047325	040246		.WORD	EMS327,EMS3
284	036016	051772	051460	051336	.WORD	EMS511,EMS504,EMS501,0
285	036026	043360	050007	040246	EMT207: .WORD	EMS54,EMS354,EMS3

286	036034	051772	051336	000000		.WORD	EMS511,EMS501,0
287	036042	043360	050007	050240	EMT210:	.WORD	EMS54,EMS354,EMS364,EMS250
288	036052	051772	051460	000000		.WORD	EMS511,EMS504,0
289	036060	043360	050007	050240	EMT211:	.WORD	EMS54,EMS354,EMS364,EMS255
290	036070	051772	051460	000000		.WORD	EMS511,EMS504,0
291	036076	047506	043435		EMT212:	.WORD	EMS337,EMS55
292	036102	051772	051460	000000		.WORD	EMS511,EMS504,0
293	036110	043513	047434	047637	EMT213:	.WORD	EMS56,EMS334,EMS345,EMS373,EMS262,EMS327,EMS251
294	036126	051772	051336			.WORD	EMS511,EMS501
295	036132	043513	047444	000000		.WORD	EMS56,EMS335,0
296	036140	047506	043513		EMT214:	.WORD	EMS337,EMS56
297	036144	051772	051336	000000		.WORD	EMS511,EMS501,0
298	036152	043606	047420	047724	EMT215:	.WORD	EMS57,EMS333,EMS350,EMS60,EMS334
299	036164	051772	051336	000000		.WORD	EMS511,EMS501,0
300	036172	050223	046622	040344	EMT216:	.WORD	EMS363,EMS306,EMS5
301	036200	051772	051336	000000		.WORD	EMS511,EMS501,0
302	036206	050223	046622	043744	EMT217:	.WORD	EMS363,EMS306,EMS61
303	036214	051772	051336	000000		.WORD	EMS511,EMS501,0
304	036222	044017	047420	050514	EMT220:	.WORD	EMS62,EMS333,EMS375,EMS251
305	036232	051772	051336	000000		.WORD	EMS511,EMS501,0
306	036240	044017	047420	050530	EMT221:	.WORD	EMS62,EMS333,EMS376,EMS262
307	036250	051772	051336	000000		.WORD	EMS511,EMS501,0
308	036256	044017	047420	050530	EMT222:	.WORD	EMS62,EMS333,EMS376,EMS250
309	036266	051772	051336	000000		.WORD	EMS511,EMS501,0
310	036274	047665	044017	047533	EMT223:	.WORD	EMS346,EMS62,EMS341,EMS603
311	036304	051772	051336	000000		.WORD	EMS511,EMS501,0
312	036312	047665	044100	050530	EMT224:	.WORD	EMS346,EMS63,EMS376,EMS262
313	036322	051772	051336	051433		.WORD	EMS511,EMS501,EMS503,0
314	036332	044100	047420	047724	EMT225:	.WORD	EMS63,EMS333,EMS350,EMS363,EMS310,EMS262
315	036346	051772	051336	000000		.WORD	EMS511,EMS501,0
316	036354	044100	050104	042337	EMT226:	.WORD	EMS63,EMS357,EMS36,EMS372
317	036364	051772	051336	000000		.WORD	EMS511,EMS501,0
318	036372	044100	050064	050126	EMT227:	.WORD	EMS63,EMS356,EMS360,EMS15
319	036402	051772	051336	000000		.WORD	EMS511,EMS501,0
320	036410	044100	050064	050154	EMT230:	.WORD	EMS63,EMS356,EMS361,EMS15
321	036420	051772	051336	000000		.WORD	EMS511,EMS501,0
322	036426	044100	050064	042477	EMT231:	.WORD	EMS63,EMS356,EMS41
323	036434	051772	051336	000000		.WORD	EMS511,EMS501,0
324	036442	042477	050544	047542	EMT232:	.WORD	EMS41,EMS377,EMS342,EMS365,EMS63,EMS332
325	036456	051772	051336	000000		.WORD	EMS511,EMS501,0
326	036464	050311	047763	050261	EMT233:	.WORD	EMS367,EMS353,EMS365,EMS63,EMS401
327	036476	051772	051433	051336		.WORD	EMS511,EMS503,EMS501,0
328	036506	041135	050544	050425	EMT234:	.WORD	EMS16,EMS377,EMS372,EMS365,EMS64,EMS354,EMS3
329	036524	051772	051433	051336		.WORD	EMS511,EMS503,EMS501,0
330	036534	046514	044210	046636	EMT235:	.WORD	EMS300,EMS65,EMS307,EMS2
331	036544	051772	051363	051433		.WORD	EMS511,EMS502,EMS503,0
332	036554	043513	050544	050530	EMT236:	.WORD	EMS56,EMS377,EMS376,EMS252,EMS372,EMS350
333	036570	044210	050572			.WORD	EMS65,EMS401
334	036574	051772	051363	051433		.WORD	EMS511,EMS502,EMS503,EMS501,0
335	036606	046514	044251	046636	EMT237:	.WORD	EMS300,EMS66,EMS307,EMS2
336	036616	051772	051336	051433		.WORD	EMS511,EMS501,EMS503,0
337	036626	041057	050555	050425	EMT240:	.WORD	EMS15,EMS400,EMS372,EMS350,EMS66,EMS401
338	036642	051772	051433	051336		.WORD	EMS511,EMS503,EMS501,0
339	036652	044251	047467	047522	EMT241:	.WORD	EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS604
340	036670	051772	051433	000000		.WORD	EMS511,EMS503,0
341	036676	050616	052223	050607	EMT242:	.WORD	EMS403,EMS604,EMS402,EMS21,EMS377
342	036710	051772	051433			.WORD	EMS511,EMS503

343	036714	042337	047444	000000		.WORD	EMS36,EMS335,0
344	036722	044251	047467	047522	EMT243:	.WORD	EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS604
345	036740	051772	051433	000000		.WORD	EMS511,EMS503,0
346	036746	044100	050572	050676	EMT244:	.WORD	EMS63,EMS401,EMS405,EMS604
347	036756	051772	051433	000000		.WORD	EMS511,EMS503,0
348	036764	042337	050361	050676	EMT245:	.WORD	EMS36,EMS370,EMS405,EMS604
349	036774	051772	051433	000000		.WORD	EMS511,EMS503,0
350	037002	050616	052223	050607	EMT246:	.WORD	EMS403,EMS604,EMS402,EMS24,EMS377
351	037014	051772	051433			.WORD	EMS511,EMS503
352	037020	042337	047444	000000		.WORD	EMS36,EMS335,0
353	037026	044326	047402	050676	EMT247:	.WORD	EMS67,EMS332,EMS405,EMS604
354	037036	051772	051433	000000		.WORD	EMS511,EMS503,0
355	037044	044251	047467	047522	EMT250:	.WORD	EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS605
356	037062	051772	051433	000000		.WORD	EMS511,EMS503,0
357	037070	050616	052250	050607	EMT251:	.WORD	EMS403,EMS605,EMS402,EMS21,EMS377
358	037102	051772	051433			.WORD	EMS511,EMS503
359	037106	042337	047444	000000		.WORD	EMS36,EMS335,0
360	037114	044251	047467	047522	EMT252:	.WORD	EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS605
361	037132	051772	051433	000000		.WORD	EMS511,EMS503,0
362	037140	044100	050572	050676	EMT253:	.WORD	EMS63,EMS401,EMS405,EMS605
363	037150	051772	051433	000000		.WORD	EMS511,EMS503,0
364	037156	042337	050361	050676	EMT254:	.WORD	EMS36,EMS370,EMS405,EMS605
365	037166	051772	051433	000000		.WORD	EMS511,EMS503,0
366	037174	050616	052250	050607	EMT255:	.WORD	EMS403,EMS605,EMS402,EMS24,EMS377
367	037206	051772	051433			.WORD	EMS511,EMS503
368	037212	042337	047444	000000		.WORD	EMS36,EMS335,0
369	037220	044326	047402	050676	EMT256:	.WORD	EMS67,EMS332,EMS405,EMS605
370	037230	051772	051433	000000		.WORD	EMS511,EMS503,0
371	037236	044251	047467	047522	EMT257:	.WORD	EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS606
372	037254	051772	051433	000000		.WORD	EMS511,EMS503,0
373	037262	050616	052266	050607	EMT260:	.WORD	EMS403,EMS606,EMS402,EMS21,EMS377
374	037274	051772	051433			.WORD	EMS511,EMS503
375	037300	042337	047444	000000		.WORD	EMS36,EMS335,0
376	037306	044251	047467	047522	EMT261:	.WORD	EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS606
377	037324	051772	051433	000000		.WORD	EMS511,EMS503,0
378	037332	044100	050572	050676	EMT262:	.WORD	EMS63,EMS401,EMS405,EMS606
379	037342	051772	051433	000000		.WORD	EMS511,EMS503,0
380	037350	042337	050361	050676	EMT263:	.WORD	EMS36,EMS370,EMS405,EMS606
381	037360	051772	051433	000000		.WORD	EMS511,EMS503,0
382	037366	050616	052266	050607	EMT264:	.WORD	EMS403,EMS606,EMS402,EMS24,EMS377
383	037400	051772	051433			.WORD	EMS511,EMS503
384	037404	042337	047444	000000		.WORD	EMS36,EMS335,0
385	037412	044440	050572	050676	EMT265:	.WORD	EMS70,EMS401,EMS405,EMS606
386	037422	051772	051433	000000		.WORD	EMS511,EMS503,0
387	037430	044326	047402	050676	EMT266:	.WORD	EMS67,EMS332,EMS405,EMS606
388	037440	051772	051433	000000		.WORD	EMS511,EMS503,0
389	037446	044251	050064	050721	EMT267:	.WORD	EMS66,EMS356,EMS407
390	037454	051772	051433	000000		.WORD	EMS511,EMS503,0
391	037462	044251	047467	047522	EMT270:	.WORD	EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS607
392	037500	051772	051433	000000		.WORD	EMS511,EMS503,0
393	037506	050616	052306	050607	EMT271:	.WORD	EMS403,EMS607,EMS402,EMS21,EMS377
394	037520	051772	051433			.WORD	EMS511,EMS503
395	037524	041741	047467	000000		.WORD	EMS27,EMS336,0
396	037532	041741	050361	050676	EMT272:	.WORD	EMS27,EMS370,EMS405,EMS607
397	037542	051772	051433	000000		.WORD	EMS511,EMS503,0
398	037550	041741	050376	050676	EMT273:	.WORD	EMS27,EMS371,EMS405,EMS607
399	037560	051772	051433	000000		.WORD	EMS511,EMS503,0

400	037566	042337	050657	050676	EMT274: .WORD	EMS36,EMS404,EMS405,EMS607
401	037576	051772	051433	000000	.WORD	EMS511,EMS503,0
402	037604	043317	050572	050676	EMT275: .WORD	EMS53,EMS401,EMS405,EMS607
403	037614	051772	051433	051363	.WORD	EMS511,EMS503,EMS502,0
404	037624	044326	047402	050676	EMT276: .WORD	EMS67,EMS332,EMS405,EMS607
405	037634	051772	051433	000000	.WORD	EMS511,EMS503,0
406	037642	044251	047467	047522	EMT277: .WORD	EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS607
407	037660	051772	051433	000000	.WORD	EMS511,EMS503,0
408	037666	050616	052306	050607	EMT300: .WORD	EMS403,EMS607,EMS402,EMS24,EMS377
409	037700	051772	051433		.WORD	EMS511,EMS503
410	037704	041741	047467	000000	.WORD	EMS27,EMS336,0
411	037712	044440	050572	050676	EMT301: .WORD	EMS70,EMS401,EMS405,EMS607
412	037722	051772	051433	000000	.WORD	EMS511,EMS503,0
413						

1	037730	052324	000000	EHT1:	.WC)	EH1,0
2	037734	052402	000000	EHT2:	.WORD	EH2,0
3	037740	052417	000000	EHT5:	.WORD	EH5,0
4	037744	052455	000000	EHT7:	.WORD	EH7,0
5	037750	052474	000000	EHT47:	.WORD	EH47,0
6	037754	052522	000000	EHT52:	.WORD	EH52,0
7	037760	052620	000000	EHT57:	.WORD	EH57,0
8	037764	052716	000000	EHT61:	.WORD	EH61,0
9	037770	052754	000000	EHT65:	.WORD	EH65,0
10	037774	053030	000000	EHT71:	.WORD	EH71,0
11	040000	052410	000000	EHT74:	.WORD	EH3,0
12	040004	053105	000000	EHT115:	.WORD	EH115,0
13	040010	053202	000000	EHT130:	.WORD	EH130,0
14	040014	053320	000000	EHT132:	.WORD	EH132,0
15	040020	053416	000000	EHT142:	.WORD	EH142,0
16	040024	053514	000000	EHT145:	.WORD	EH145,0
17	040030	053631	000000	EHT150:	.WORD	EH150,0
18	040034	053727	000000	EHT213:	.WORD	EH213,0
19	040040	054024	000000	EHT220:	.WORD	EH220,0
20						

1	040044	054062	EDT1:	.WORD	ED1
2	040046	054072	EDT2:	.WORD	ED2
3	040050	054076	EDT5:	.WORD	ED5
4	040052	054104	EDT47:	.WORD	ED47
5	040054	054114	EDT52:	.WORD	ED52
6	040056	054126	EDT57:	.WORD	ED57
7	040060	054140	EDT61:	.WORD	ED61
8	040062	054152	EDT65:	.WORD	ED65
9	040064	054162	EDT71:	.WORD	ED71
10	040066	054072	EDT74:	.WORD	ED2
11	040070	054172	EDT115:	.WORD	ED115
12	040072	054204	EDT130:	.WORD	ED130
13	040074	054172	EDT132:	.WORD	ED115
14	040076	054220	EDT220:	.WORD	ED220
15					

1	040100	054224	EFT1:	.WORD	EF1
2	040102	054227	EFT2:	.WORD	EF2
3	040104	054230	EFT5:	.WORD	EF5
4	040106	054232	EFT57:	.WORD	EF57
5	040110	054224	EFT65:	.WORD	EF1
6	040112	054224	EFT71:	.WORD	EF1
7	040114	054227	EFT74:	.WORD	EF2
8	040116	054232	EFT115:	.WORD	EF57
9	040120	054236	EFT130:	.WORD	EF130
10	040122	054232	EFT132:	.WORD	EF57
11	040124	054230	EFT220:	.WORD	EF5
12					

1	040126	116	117	116	EMS1:	.ASCIZ @NONEXISTENT DEVICE 'NED' (RMCS2,BIT 12) a
2	040177	103	117	116	EMS2:	.ASCIZ @CONTROLLER CLEAR 'CLR' (RMCS2,BIT 05) a
3	040246	106	125	116	EMS3:	.ASCIZ @FUNCTION CODE (RMCS1, BITS 01 - 05) a
4	040313	125	116	125	EMS4:	.ASCIZ @UNUSED BIT POSITIONS OF a
5	040344	104	105	126	EMS5:	.ASCIZ @DEVICE AVAILABLE 'DVA' (RMCS1, BIT 11) a
6	040414	120	101	122	EMS6:	.ASCIZ @PARTIY ERROR 'PAR' (RMER1, BIT 03) a
7	040460	104	101	124	EMS7:	.ASCIZ @DATA PARITY ERROR 'DPE' (RMER2, BIT 03) a
8	040531	120	101	122	EMS10:	.ASCIZ @PARITY TEST 'PAT' (RMCS2, BIT 04) a
9	040574	115	101	123	EMS11:	.ASCII @MASSBUS CONTROL BUS PARITY ERROR 'MCPE' a
10	040644	050	122	115	.ASCIZ @ (RMCS1, BIT 13) a	
11	040665	111	114	114	EMS12:	.ASCIZ @ILLEGAL REGISTER ERROR 'ILR' (RMER1, BIT 01) a
12	040743	104	111	101	EMS13:	.ASCIZ @DIAGNOSTIC MODE 'DMD' (RMMR1, BIT 00) a
13	041012	115	105	104	EMS14:	.ASCIZ @MEDIUM ON LINE 'MOL' (RMDS, BIT 12) a
14	041057	115	101	111	EMS15:	.ASCIZ @MAINTENANCE UNIT READY 'MUR' (RMMR1, BIT 09) a
15	041135	115	101	111	EMS16:	.ASCIZ @MAINTENANCE WRITE PROTECT 'MWP' (RMMR1, BIT 03) a
16	041216	127	122	111	EMS17:	.ASCIZ @WRITE LOCK 'WRL' (RMDS, BIT 11) a
17	041257	104	105	126	EMS20:	.ASCIZ @DEVICE CHECK 'DVC' (RMER2, BIT 07) a
18	041323	115	101	111	EMS21:	.ASCIZ @MAINTENANCE DRIVE FAULT 'MDF' (RMMR1, BIT 06) a
19	041402	125	116	123	EMS22:	.ASCIZ @UNSAFE STATUS 'UNS' (RMER1, BIT 14) a
20	041447	123	105	105	EMS23:	.ASCIZ @SEEK INCOMPLETE STATUS 'SKI' (RMER2, BIT 14) a
21	041525	115	101	111	EMS24:	.ASCIZ @MAINTENANCE SEEK ERROR 'MSER' (RMMR1, BIT 07) a
22	041604	120	117	123	EMS25:	.ASCIZ @POSITIONING IN PROGRESS 'PIP' (RMDS, BIT 13) a
23	041662	115	101	111	EMS26:	.ASCIZ @MAINTENANCE ON CYLINDER 'MOC' (RMMR1, BIT 08) a
24	041741	105	116	104	EMS27:	.ASCIZ @END OF BLOCK 'EBL' (RMMR1, BIT 13) a
25	042005	104	111	101	EMS30:	.ASCIZ @DIAGNOSTIC END OF BLOCK 'DEBL' (RMMR1, BIT 13) a
26	042065	114	101	123	EMS31:	.ASCIZ @LAST SECTOR STATUS 'LS' (RMMR1, BIT 02) a
27	042136	114	101	123	EMS32:	.ASCIZ @LAST SECTOR/TRACK STATUS 'LST' (RMMR1, BIT 01) a
28	042216	123	105	103	EMS33:	.ASCIZ @SECTOR ADDRESS BITS OF a
29	042246	124	122	101	EMS34:	.ASCIZ @TRACK ADDRESS BITS OF a
30	042275	126	117	114	EMS35:	.ASCIZ @VOLUME VALID 'VV' (RMDS, BIT 06) a
31	042337	107	117	040	EMS36:	.ASCIZ @GO BIT (RMCS1, BIT 00) a
32	042367	103	131	114	EMS37:	.ASCIZ @CYLINDER ADDRESS BITS OF a
33	042421	114	101	123	EMS40:	.ASCIZ @LAST BLOCK TRANSFERRED, 'LBT' (RMDS, BIT 10) a
34	042477	103	117	115	EMS41:	.ASCIZ @COMPOSITE ERROR 'ERR' (RMDS, BIT 14) a
35	042545	103	117	115	EMS42:	.ASCIZ @COMMAND SEQUENCER TEST BIT 'TST' (RMMR2, BIT 12) a
36	042627	104	122	111	EMS43:	.ASCIZ @DRIVE READY STATUS 'DRY' (RMDS, BIT 07) a
37	042700	103	117	116	EMS44:	.ASCIZ @CONTINUE 'CONT' (RMMR1, BIT 06) a
38	042741	111	116	126	EMS45:	.ASCIZ @INVALID COMMAND ERROR 'IVC' (RMER2, BIT 12) a
39	043016	114	117	123	EMS46:	.ASCIZ @LOSS OF SYSTEM CLOCK ERROR 'LSC' (RMER2, BIT 11) a
40	043100	117	103	103	EMS47:	.ASCIZ @OCCUPIED 'OCC' (RMMR1, BIT 15) a
41	043140	111	114	114	EMS50:	.ASCIZ @ILLEGAL FUNCTION 'ILF' (RMER1, BIT 0) a
42	043207	117	106	106	EMS51:	.ASCIZ @OFFSET DIRECTION 'OFD' (RMOF, BIT 07) a
43	043256	117	106	106	EMS52:	.ASCIZ @OFFSET MODE 'OM' (RMDS, BIT 00) a
44	043317	122	125	116	EMS53:	.ASCIZ @RUN AND GO 'RG' (RMMR1, BIT 14) a
45	043360	111	116	126	EMS54:	.ASCIZ @INVALID ADDRESS ERROR 'IAE' (RMER1, BIT 10) a
46	043435	101	104	104	EMS55:	.ASCIZ @ADDRESS OVERFLOW ERROR 'AOE' (RMER1, BIT 09) a
47	043513	122	105	107	EMS56:	.ASCII @REGISTER MODIFICATION REFUSED ERROR a
48	043557	042	122	115	.ASCIZ @ 'RMR' (RMER1, BIT 02) a	
49	043606	104	122	111	EMS57:	.ASCIZ @DRIVE REQUEST REQUIRED STATUS 'DRQ' (RMDT, BIT 11) a
50	043672	120	122	117	EMS60:	.ASCIZ @PROGRAMMABLE STATUS 'PGM' (RMDS, BIT 09) a
51	043744	104	122	111	EMS61:	.ASCIZ @DRIVE PRESENT STATUS 'DPR' (RMDS, BIT 08) a
52	044017	120	117	122	EMS62:	.ASCIZ @SPORT REQUEST FLOP 'RQA,RQB' (RMMR2, BITS 15,14) a
53	044100	101	124	124	EMS63:	.ASCIZ @ATTENTION 'ATA' (RMDS, BIT 15) a
54	044140	127	122	111	EMS64:	.ASCIZ @WRITE LOCK ERROR 'WLE' (RMER1, BIT 11) a
55	044210	105	130	103	EMS65:	.ASCIZ @EXCEPTION 'REX' (RMMR1, BIT 12) a
56	044251	111	116	126	EMS66:	.ASCIZ @INVALID COMMAND ERROR 'IVC' (RMER2, BIT 12) a
57	044326	124	101	107	EMS67:	.ASCIZ @TAG BUS (RMMR2, BITS 00-09) OR TAG CONTROL a

58	044402	114	111	116		.ASCIZ	@LINES (RMMR2, BITS 10,11,13) @
59	044440	123	105	101	EMS70:	.ASCIZ	@SEARCH ENABLE 'ESRC' (RMMR1, BIT 11) @
60							
61	044506	117	120	105	EMS71:	.ASCIZ	@OPERATION INCOMPLETE ERROR 'DPI'(RMR1 BIT13) @
62	044565	104	111	123	EMS72:	.ASCIZ	@DISABLE SEARCH TIMEOUT 'DTO' (RMMR1 BIT 12) @
63	044642	104	122	111	EMS73:	.ASCIZ	@DRIVE TIMING ERROR 'DTE'(RMR1,BIT 12) @
64	044712	115	101	111	EMS74:	.ASCIZ	@MAINTENANCE INDEX PULSE 'MI'(RMMR1 BIT 2) @
65	044765	115	101	111	EMS75:	.ASCIZ	@MAINTENANCE SECTOR PULSE 'MS'(RMMR1 BIT 5) @
66	045041	106	117	122	EMS76:	.ASCIZ	@FORMAT BIT 'FMT16' (RMOF,BIT12) @
67	045102	123	105	103	EMS77:	.ASCIZ	@SECTOR COMPARE 'MSC'(RMMR1 BIT 1) @
68	045145	120	122	117	EMS100:	.ASCIZ	@PROM STROBE 'WC' (RMMR1 BIT 5) @
69	045205	115	101	111	EMS101:	.ASCIZ	@MAINTENANCE BIT CLOCK 'MCLK' (RMMR1 BIT11) @
70	045261	114	117	117	EMS102:	.ASCIZ	@LOOKING FOR SYNC 'PLFS'(RMMR1,BIT10) @
71	045327	127	122	111	EMS103:	.ASCIZ	@WRITE GATE 'BB00' (RMMR2 BIT0) @
72	045367	110	105	101	EMS104:	.ASCIZ	@HEADER SYNC PATTERN @
73	045414	110	105	101	EMS105:	.ASCIZ	@HEADER AREA 'PHA' (RMMR1 BIT 7) @
74	045455	105	116	101	EMS106:	.ASCIZ	@ENABLE CRC OUT 'ECRC' (RMMR1 BIT 9) @
75	045522	110	105	101	EMS107:	.ASCIZ	@HEADER @
76	045532	122	105	101	EMS110:	.ASCIZ	@READ GATE 'BB01'(RMMR2 BIT 1) @
77	045571	110	105	101	EMS111:	.ASCIZ	@HEADER ERROR @
78	045607	104	101	124	EMS112:	.ASCIZ	@DATA TIMING SEQUENCER OUTPUT @
79	045645	104	101	124	EMS113:	.ASCIZ	@DATA AREA 'PDA' (RMMR1 BIT 8) @
80	045704	105	116	101	EMS114:	.ASCIZ	@ENABLE ECC OUT 'EECC' (RMMR1, BIT 4) @
81	045752	105	103	103	EMS115:	.ASCIZ	@ECC PATTERN @
82	045767	105	103	103	EMS116:	.ASCIZ	@ECC ERROR @
83	046002	104	111	123	EMS250:	.ASCIZ	@DISK ADDRESS REGISTER (RMDA) @
84	046040	103	117	116	EMS251:	.ASCIZ	@CONTROL STATUS REGISTER #1 (RMCS1) @
85	046104	105	122	122	EMS252:	.ASCIZ	@ERROR REGISTER #1 (RMR1) @
86	046137	105	122	122	EMS253:	.ASCIZ	@ERROR REGISTER #2 (RMR2) @
87	046172	115	101	111	EMS254:	.ASCIZ	@MAINTENANCE REGISTER #1 (RMMR1) @
88	046233	104	105	123	EMS255:	.ASCIZ	@DESIRED CYLINDER REGISTER (RMDC) @
89	046275	117	106	106	EMS256:	.ASCIZ	@OFFSET REGISTER (RMOF) @
90	046325	104	122	111	EMS257:	.ASCIZ	@DRIVE TYPE REGISTER (RMDT) @
91	046361	110	117	114	EMS260:	.ASCIZ	@HOLDING REGISTER (RMR) @
92	046412	123	105	122	EMS261:	.ASCIZ	@SERIAL NUMBER REGISTER (RMSN) @
93	046451	101	124	124	EMS262:	.ASCIZ	@ATTENTION SUMMARY REGISTER (RMAS) @
94							
95	046514	103	101	116	EMS300:	.ASCIZ	@CANNOT CLEAR @
96	046532	103	101	116	EMS301:	.ASCIZ	@CANNOT WRITE/READ @
97	046555	101	116	131	EMS302:	.ASCIZ	@ANY DEVICE REGISTER @
98	046602	127	111	124	EMS303:	.ASCIZ	@WITHOUT @
99	046613	105	122	122	EMS304:	.ASCIZ	@ERROR @
100	046622	101	040	117	EMS306:	.ASCIZ	@A ONE FROM @
101	046636	125	123	111	EMS307:	.ASCIZ	@USING MASSBUS INITIALIZE, I.E., @
102	046677	101	040	132	EMS310:	.ASCIZ	@A ZERO FROM @
103	046714	105	126	105	EMS311:	.ASCIZ	@EVERY DEVICE REGISTER BIT POSITION @
104	046760	124	110	105	EMS312:	.ASCIZ	@THE FOLLOWING BITS ARE STUCK @
105	047016	101	040	123	EMS313:	.ASCIZ	@A SHIFTING ONE BIT FROM @
106	047047	101	120	120	EMS314:	.ASCIZ	@APPEARS STUCK AT ZERO @
107	047076	101	120	120	EMS315:	.ASCIZ	@APPEARS STUCK AT ONE @
108	047124	122	105	107	EMS316:	.ASCIZ	@REGISTER SELECT @
109	047145	061	040	050	EMS317:	.ASCIZ	@1 (1,2,4,8,16) @
110	047165	062	040	050	EMS320:	.ASCIZ	@2 (1,2,4,8,16) @
111	047205	064	040	050	EMS321:	.ASCIZ	@4 (1,2,4,8,16) @
112	047225	070	040	050	EMS322:	.ASCIZ	@8 (1,2,4,8,16) @
113	047245	101	114	114	EMS323:	.ASCIZ	@ALL ONES FROM @
114	047264	101	114	114	EMS324:	.ASCIZ	@ALL ZEROS FROM @

115	047304	101	124	040	EMS325:	.ASCIZ	@AT ZERO @
116	047315	101	124	040	EMS326:	.ASCIZ	@AT ONE @
117	047325	054	040	117	EMS327:	.ASCIZ	@, OR @
118	047333	015	012	103	EMS330:	.ASCIZ	<CR><LF>@CS MBA CLRL @
119	047352	103	101	116	EMS331:	.ASCIZ	@CANNOT READ ZEROS FROM @
120	047402	111	123	040	EMS332:	.ASCIZ	@IS INCORRECT @
121	047420	111	123	040	EMS333:	.ASCIZ	@IS NOT SET @
122	047434	111	123	040	EMS334:	.ASCIZ	@IS SET @
123	047444	123	110	117	EMS335:	.ASCIZ	@SHOULD NOT BE SET @
124	047467	123	110	117	EMS336:	.ASCIZ	@SHOULD BE SET @
125	047506	103	101	116	EMS337:	.ASCIZ	@CANNOT SET @
126	047522	102	105	103	EMS340:	.ASCIZ	@BECAUSE @
127	047533	125	123	111	EMS341:	.ASCIZ	@USING @
128	047542	104	125	122	EMS342:	.ASCIZ	@DURING REGISTER TRANSFER @
129	047574	125	116	105	EMS343:	.ASCIZ	@UNEXPECTED @
130	047610	102	125	123	EMS344:	.ASCIZ	@BUS TIMEOUT (04 TRAP) @
131	047637	102	131	040	EMS345:	.ASCIZ	@BY REGISTER TRANSFER @
132	047665	103	101	116	EMS346:	.ASCIZ	@CANNOT RESET @
133	047703	127	111	124	EMS347:	.ASCIZ	@WITHOUT SETTING @
134	047724	102	125	124	EMS350:	.ASCIZ	@BUT @
135	047731	127	101	123	EMS351:	.ASCIZ	@WAS RESET BY @
136	047747	127	101	123	EMS352:	.ASCIZ	@WAS SET BY @
137	047763	111	116	040	EMS353:	.ASCIZ	@IN DIAGNOSTIC MODE @
138	050007	111	123	040	EMS354:	.ASCIZ	@IS INCORRECT ACCORDING TO @
139	050042	103	101	116	EMS355:	.ASCIZ	@CANNOT INCREMENT @
140	050064	127	101	123	EMS356:	.ASCIZ	@WAS NOT SET BY @
141	050104	127	101	123	EMS357:	.ASCIZ	@WAS NOT RESET BY @
142	050126	060	040	124	EMS360:	.ASCIZ	@0 TO 1 TRANSITION OF @
143	050154	061	040	124	EMS361:	.ASCIZ	@1 TO 0 TRANSITION OF @
144	050202	111	123	040	EMS362:	.ASCIZ	@IS INCONSISTENT @
145	050223	103	101	116	EMS363:	.ASCIZ	@CANNOT READ @
146	050240	124	105	123	EMS364:	.ASCIZ	@TEST PATTERN IN @
147	050261	101	116	104	EMS365:	.ASCIZ	@AND @
148	050266	103	101	116	EMS366:	.ASCIZ	@CANNOT INITIALIZE @
149	050311	124	110	105	EMS367:	.ASCIZ	@THE COMMAND SEQUENCER HAS BEEN CLOCKED @
150	050361	122	105	123	EMS370:	.ASCIZ	@RESET EARLY @
151	050376	104	111	104	EMS371:	.ASCIZ	@DID NOT RESET ON TIME @
152	050425	104	125	122	EMS372:	.ASCIZ	@DURING COMMAND EXECUTION @
153	050457	124	117	040	EMS373:	.ASCIZ	@TO @
154	050463	127	111	124	EMS374:	.ASCIZ	@WITH ANY COMBINATION OF @
155	050514	102	131	040	EMS375:	.ASCIZ	@BY READING @
156	050530	102	131	040	EMS376:	.ASCIZ	@BY WRITING @
157	050544	127	101	123	EMS377:	.ASCIZ	@WAS SET @
158	050555	127	101	123	EMS400:	.ASCIZ	@WAS NOT SET @
159	050572	104	111	104	EMS401:	.ASCIZ	@DID NOT SET @
160	050607	127	110	111	EMS402:	.ASCIZ	@WHILE @
161	050616	103	117	115	EMS403:	.ASCIZ	@COMMAND SEQUENCER DID NOT ABORT @
162	050657	127	101	123	EMS404:	.ASCIZ	@WAS NOT RESET @
163	050676	104	125	122	EMS405:	.ASCIZ	@DURING @
164	050706	127	101	123	EMS406:	.ASCIZ	@WAS RESET @
165	050721	123	105	101	EMS407:	.ASCIZ	@SEARCH TIMEOUT @
166	050741	101	106	124	EMS410:	.ASCIZ	@AFTER @
167	050750	127	101	123	EMS411:	.ASCIZ	@WAS CHANGED @
168	050765	123	105	124	EMS412:	.ASCIZ	@SET EARLY @
169	051000	111	116	103	EMS413:	.ASCIZ	@INCORRECT @
170	051013	103	101	116	EMS414:	.ASCIZ	@CANNOT DETECT @
171	051032	127	101	123	EMS415:	.ASCIZ	@WAS SIMULATED @

1	054062	001140	001142	001136	ED1:	.WORD	\$GDDAT,\$BDDAT,\$BDADR,0
2	054072	001136	000000		ED2:	.WORD	\$BDADR,0
3	054076	001140	001142	000000	ED5:	.WORD	\$GDDAT,\$BDDAT,0
4	054104	001174	001176	001200	ED47:	.WORD	\$TMP0,\$TMP1,\$TMP2,0
5	054114	001174	054244	001176	ED52:	.WORD	\$TMP0,BUFFER,\$TMP1,BUFFER+2,0
6	054126	001174	001176	001142	ED57:	.WORD	\$TMP0,\$TMP1,\$BDDAT,\$BDADR,0
7	054140	001200	001202	001174	ED61:	.WORD	\$TMP2,\$TMP3,\$TMP0,\$TMP1,0
8	054152	001140	001142	001174	ED65:	.WORD	\$GDDAT,\$BDDAT,\$TMP0,0
9	054162	001140	001142	001444	ED71:	.WORD	\$GDDAT,\$BDDAT,RMHRO,0
10	054172	001140	001142	001136	ED115:	.WORD	\$GDDAT,\$BDDAT,\$BDADR,\$TMP0,0
11	054204	001140	001142	001136	ED130:	.WORD	\$GDDAT,\$BDDAT,\$BDADR,\$TMP0,\$TMP1,0
12	054220	001142	001136		ED220:	.WORD	\$BDDAT,\$BDADR
13							
14	054224	000	000	000	EF1:	.BYTE	0,0,0
15	054227	000			EF2:	.BYTE	0
16	054230	000	000		EF5:	.BYTE	0,0
17	054232	000	000	000	EF57:	.BYTE	0,0,0,0
18	054236	000	000	000	EF130:	.BYTE	0,0,0,0,0
19							
20					.EVEN		
21	054244				BUFFER:		
22	054244				BUFONE:	.BLKW	258.
23	055250				BUFTWO:	.BLKW	258.
24		054244			.=BUFFER		
25							
26	054244				HELP:		
27	054244	200			.ASCII	<CRLF>	
28	054245	200			.ASCII	<CRLF>	
29	054246	114	111	123	.ASCII	@LIST OF TESTS@<CRLF>	
30	054264	055	055	055	.ASCII	@-----@<CRLF>	
31	054302	124	061	011	.ASCII	@T1 TRANSFER TEST@<CRLF>	
32	054323	124	062	011	.ASCII	@T2 CTOD TEST@<CRLF>	
33	054340	124	063	011	.ASCII	@T3 MASSBUS INITIALIZE TEST@<CRLF>	
34	054373	124	064	011	.ASCII	@T4 CLEAR STUCK ACTIVE TEST@<CRLF>	
35	054426	124	065	011	.ASCII	@T5 TRISTATE TRANSFER TEST@<CRLF>	
36	054460	124	066	011	.ASCII	@T6 REGISTER SELECT TEST@<CRLF>	
37	054510	124	067	011	.ASCII	@T7 DRIVE TYPE TEST@<CRLF>	
38	054533	124	061	060	.ASCII	@T10 DEVICE AVAILABLE TEST@<CRLF>	
39	054565	124	061	061	.ASCII	@T11 SEARCH TIMEOUT TEST@<CRLF>	
40	054615	124	061	062	.ASCII	@T12 SET DTE TEST@<CRLF>	
41	054636	124	061	063	.ASCII	@T13 FORMAT CHANGE TEST@<CRLF>	
42	054665	124	061	064	.ASCII	@T14 PROM STROBE TEST@<CRLF>	
43	054712	124	061	065	.ASCII	@T15 SYNC WORD COUNT INHIBIT TEST@<CRLF>	
44	054753	124	061	066	.ASCII	@T16 SYNC DETECTION TEST@<CRLF>	
45	055003	124	061	067	.ASCII	@T17 ABORT SYNC DETECTION TEST@<CRLF>	
46	055041	124	062	060	.ASCII	@T20 SYNC GENERATION TEST@<CRLF>	
47	055072	124	062	061	.ASCII	@T21 WRITE HEADER TEST@<CRLF>	
48	055120	124	062	062	.ASCII	@T22 HEADER COMPARE TEST @<CRLF>	
49	055151	124	062	063	.ASCII	@T23 ECC GENERATION TEST@<CRLF>	
50	055201	124	062	064	.ASCII	@T24 ECC DETECTION TEST@<CRLF>	
51	055230	200			.ASCII	<CRLF>	
52	055231	117	120	105	.ASCII	@OPERATIONAL SWITCH SETTINGS@<CRLF>	
53	055265	055	055	055	.ASCII	@-----@<CRLF>	
54	055321	123	127	111	.ASCII	@SWITCH USE@<CRLF>	
55	055345	055	055	055	.ASCII	@-----@<CRLF>	
56	055402	040	040	061	.ASCII	@ 15 HALT ON ERROR@<CRLF>	
57	055426	040	040	061	.ASCII	@ 14 LOOP ON TEST@<CRLF>	

58	055451	040	040	061	.ASCII	@	13	INHIBIT ERROR TYPEOUTS@<CRLF>
59	055506	040	040	061	.ASCII	@	12	@<CRLF>
60	055515	040	040	061	.ASCII	@	11	INHIBIT ITERATIONS@<CRLF>
61	055546	040	040	061	.ASCII	@	10	BELL ON ERROR@<CRLF>
62	055572	040	040	040	.ASCII	@	9	LOOP ON ERROR@<CRLF>
63	055616	040	040	040	.ASCII	@	8	LOOP ON TEST IN SWR<7:0>@<CRLF>
64	055655	040	040	040	.ASCII	@	7	TN128@<CRLF>
65	055671	040	040	040	.ASCII	@	6	TN64@<CRLF>
66	055704	040	040	040	.ASCII	@	5	TN32@<CRLF>
67	055717	040	040	040	.ASCII	@	4	TN16@<CRLF>
68	055732	040	040	040	.ASCII	@	3	TN8@<CRLF>
69	055744	040	040	040	.ASCII	@	2	TN4@<CRLF>
70	055756	040	040	040	.ASCII	@	1	TN2@<CRLF>
71	055770	040	040	040	.ASCII	@	0	TN1@<CRLF>
72								
73	000200			.END			200	

ABASE = 176700	AVECT1= 120254	CNSL00 031176	ED1 054062	EH220 054024
ACDW1 = 000000	AVECT2= 000000	CNSL01 031231	ED115 054172	EH3 052410
ACDW2 = 000000	A16 = 000400	CNSL02 031247	ED130 054204	EH47 052474
ACPUOP= 000000	A17 = 001000	CNSL03 031311	ED2 054072	EH5 052417
ADDW0 = 000000	BADTMO 002636	CNSL04 031331	ED220 054220	EH52 052522
ADDW1 = 000000	BAI = 000010	CNSL05 031365	ED47 054104	EH57 052620
ADDW10= 000000	BB00 = 000001	CNSL06 031377	ED5 054076	EH61 052716
ADDW11= 000000	BB01 = 000002	CNSL07 031430	ED52 054114	EH65 05 754
ADDW12= 000000	BB02 = 000004	CNSL08 031573	ED57 054126	EH7 052455
ADDW13= 000000	BB03 = 000010	CNSL09 031574	ED61 054140	EH71 053030
ADDW14= 000000	BB04 = 000020	CNTCLR 023334	ED65 054152	EMS1 040126
ADDW15= 000000	BB05 = 000040	COMMA 031107	ED71 054162	EMS10 040531
ADDW2 = 000000	BB06 = 000100	CONT = 000100	EECC = 000020	EMS100 045145
ADDW3 = 000000	BB07 = 000200	CR = 000015	EFT1 040100	EMS101 045205
ADDW4 = 000000	BB08 = 000400	CRLF = 000200	EFT115 040116	EMS102 045261
ADDW5 = 000000	BB09 = 001000	CYLMSK= 001777	EFT130 040120	EMS103 045327
ADDW6 = 000000	BIT0 = 000001	DBCK = 100000	EFT132 040122	EMS104 045367
ADDW7 = 000000	BIT00 = 000001	DBEN = 040000	EFT2 040102	EMS105 045414
ADDW8 = 000000	BIT01 = 000002	DBL = 002000	EFT220 040124	EMS106 045455
ADDW9 = 000000	BIT02 = 000004	DCK = 100000	EFT5 040104	EMS107 045522
ADEVCT= 000000	BIT03 = 000010	DDISP = 177570	EFT57 040106	EMS11 040574
ADEVN = 000000	BIT04 = 000020	DEBL = 020000	EFT65 040110	EMS110 045532
ADR = 000001	BIT05 = 000040	DISPLA 001156	EFT71 040112	EMS111 045571
AENV = 000000	BIT06 = 000100	DISPRE 000174	EFT74 0401 1	EMS112 045607
AENVM = 000000	BIT07 = 000200	DLT = 100000	EF1 054224	EMS113 045645
AFATAL= 000000	BIT08 = 000400	DMD = 000001	EF130 054236	EMS114 045704
ALL 031076	BIT09 = 001000	DPE = 000010	EF2 054227	EMS115 045752
AMADR1= 000000	BIT1 = 000002	DPEHI = 040000	EF5 054230	EMS116 045767
AMADR2= 000000	BIT10 = 002000	DPELO = 020000	EF57 054232	EMS12 040665
AMADR3= 000000	BIT11 = 004000	DPR = 000400	EHT1 037730	EMS13 0407 3
AMADR4= 000000	BIT12 = 010000	DRQ = 004000	EHT115 040004	EMS14 041012
AMAMS1= 000000	BIT13 = 020000	DRVCLR= 000010	EHT130 040010	EMS15 041057
AMAMS2= 000000	BIT14 = 040000	DRY = 000200	EHT132 04C014	EMS16 041135
AMAMS3= 000000	BIT15 = 100000	DSWR = 177570	EHT142 040020	EMS17 041216
AMAMS4= 000000	BIT2 = 000004	DTE = 010000	EHT145 040024	EMS2 040177
AMSGAD= 000000	BIT3 = 000010	DTO = 010000	EHT150 040030	EMS20 041257
AMSGLG= 000000	BIT4 = 000020	DULPRT= 024024	EHT2 037734	EMS21 041323
AMCGTY= 000000	BIT5 = 000040	DVA = 004000	EHT213 040034	EMS22 041402
AMTYP1= 000000	BIT6 = 000100	DVC = 000200	EHT220 040040	EMS23 041447
AMTYP2= 000000	BIT7 = 000200	EBL = 020000	EHT47 037750	EMS24 041525
AMTYP3= 000000	BIT8 = 000400	ECH = 000100	EHT5 037740	EMS25 041604
AMTYP4= 000000	BIT9 = 001000	ECL = 004000	EHT52 037754	EMS250 046002
AOE = 001000	BOTADR 022412	ECRC = 001000	EHT57 037760	EMS251 046040
APASS = 000000	BOTFLG 022414	EDT1 040044	EHT61 037764	EMS252 046104
APE = 100000	BPTVEC= 000014	EDT115 040070	EHT65 037770	EMS253 046137
APRIOR= 000000	BSE = 100000	EDT130 0400 7	EHT7 037744	EMS254 046172
APTCSE= 000040	BUFFER 054244	EDT132 040074	EHT71 037774	EMS255 046233
APTENV= 000001	BUF ONE 054244	EDT2 040046	EHT74 040000	EMS256 046275
APTSIZ= 000200	BUFTWO 055250	EDT220 040076	EH1 052324	EMS257 046325
APTSPO= 000100	CC = 004000	EDT47 040052	EH115 053105	EMS26 041662
ASWREG= 000000	CH = 002000	EDT5 040050	EH130 053202	EMS260 046361
ATA = 100000	CHRCNT 022415	EDT52 040054	EH132 053320	EMS261 046412
ATESTN= 000000	CKSWR = 104410	EDT57 040056	EH142 053416	EMS262 046451
ATMSK= 000377	CLKSNC 024572	EDT61 040060	EH145 053514	EMS27 041741
ATNTBL 032014	CLOCK 001532	EDT65 040062	EH150 053631	EMS3 040246
AUNIT = 000000	CLR = 000040	EDT71 040064	EH2 052402	EMS30 042005
AUSWR = 000000	CMNSTA 005070	EDT74 040066	EH213 053727	EMS300 046514

EMS301	046532	EMS365	050261	EMS56	043513	EMT13	032264	EMT211	036060
EMS302	046555	EMS366	050266	EMS57	043606	EMT130	034510	EMT212	036076
EMS303	046602	EMS367	050311	EMS6	040414	EMT131	034526	EMT213	036110
EMS304	046613	EMS37	042367	EMS60	043672	EMT132	034544	EMT214	036140
EMS306	046622	EMS370	050361	EMS600	052071	EMT133	034564	EMT215	036152
EMS307	046636	EMS371	050376	EMS601	052121	EMT134	034604	EMT216	036172
EMS31	042065	EMS372	050425	EMS602	052141	EMT135	034620	EMT217	036206
EMS310	046677	EMS373	050457	EMS603	052202	EMT136	034636	EMT22	032436
EMS311	046714	EMS374	050463	EMS604	052223	EMT137	034650	EMT220	036222
EMS312	046760	EMS375	050514	EMS605	052250	EMT14	032302	EMT221	036240
EMS313	047016	EMS376	050530	EMS606	052266	EMT140	034666	EMT222	036256
EMS314	047047	EMS377	050544	EMS607	052306	EMT141	034706	EMT223	036274
EMS315	047076	EMS4	040313	EMS61	043744	EMT142	034722	EMT224	036312
EMS316	047124	EMS40	042421	EMS62	044017	EMT143	034740	EMT225	036332
EMS317	047145	EMS400	050555	EMS63	044100	EMT144	034754	EMT226	036354
EMS32	042136	EMS401	050572	EMS64	044140	EMT145	034776	EMT227	036372
EMS320	047165	EMS402	050607	EMS65	044210	EMT146	035020	EMT23	032462
EMS321	047205	EMS403	050616	EMS66	044251	EMT147	035036	EMT230	036410
EMS322	047225	EMS404	050657	EMS67	044326	EMT15	032320	EMT231	036426
EMS323	047245	EMS405	050676	EMS7	040460	EMT150	035050	EMT232	036442
EMS324	047264	EMS406	050706	EMS70	044440	EMT151	035072	EMT233	036464
EMS325	047304	EMS407	050721	EMS71	044506	EMT152	035104	EMT234	036506
EMS326	047315	EMS41	042477	EMS72	044565	EMT153	035120	EMT235	036534
EMS327	047325	EMS410	050741	EMS73	044642	EMT154	035140	EMT236	036554
EMS33	042216	EMS411	050750	EMS74	044712	EMT155	035160	EMT237	036606
EMS330	047333	EMS412	050765	EMS75	044765	EMT156	035176	EMT24	032506
EMS331	047352	EMS413	051000	EMS76	045041	EMT157	035220	EMT240	036626
EMS332	047402	EMS414	051013	EMS77	045102	EMT16	032336	EMT241	036652
EMS333	047420	EMS415	051032	EMTVEC=	000030	EMT160	035236	EMT242	036676
EMS334	047434	EMS416	051051	EMT1	032024	EMT161	035266	EMT243	036722
EMS335	047444	EMS417	051062	EMT10	032212	EMT162	035304	EMT244	036746
EMS336	047467	EMS42	042545	EMT100	033662	EMT163	035322	EMT245	036764
EMS337	047506	EMS420	051100	EMT101	033702	EMT164	035344	EMT246	037002
EMS34	042246	EMS421	051113	EMT102	033722	EMT165	035366	EMT247	037026
EMS340	047522	EMS422	051130	EMT103	033734	EMT166	035412	EMT25	032532
EMS341	047533	EMS43	042627	EMT104	033760	EMT167	035432	EMT250	037044
EMS342	047542	EMS44	042700	EMT105	034000	EMT17	032354	EMT251	037070
EMS343	047574	EMS45	042741	EMT106	034020	EMT170	035454	EMT252	037114
EMS344	047610	EMS46	043016	EMT107	034044	EMT171	035466	EMT253	037140
EMS345	047637	EMS47	043100	EMT11	032230	EMT172	035502	EMT254	037156
EMS346	047665	EMS5	040344	EMT110	034064	EMT173	035520	EMT255	037174
EMS347	047703	EMS50	043140	EMT111	034106	EMT174	035542	EMT256	037220
EMS35	042275	EMS500	051176	EMT112	034126	EMT175	035570	EMT257	037236
EMS350	047724	EMS501	051336	EMT113	034146	EMT176	035606	EMT26	032552
EMS351	047731	EMS502	051363	EMT114	034172	EMT177	035624	EMT260	037262
EMS352	047747	EMS503	051433	EMT115	034214	EMT2	032032	EMT261	037306
EMS353	047763	EMS504	051460	EMT116	034240	EMT20	032372	EMT262	037332
EMS354	050007	EMS505	051513	EMT117	034260	EMT200	035642	EMT263	037350
EMS355	050042	EMS506	051566	EMT12	032246	EMT201	035660	EMT264	037366
EMS356	050064	EMS507	051627	EMT120	034300	EMT202	035676	EMT265	037412
EMS357	050104	EMS51	043207	EMT121	034324	EMT203	035714	EMT266	037430
EMS36	042337	EMS510	051717	EMT122	034344	EMT204	035736	EMT267	037446
EMS360	050126	EMS511	051772	EMT123	034364	EMT205	035756	EMT27	032604
EMS361	050154	EMS52	043256	EMT124	034410	EMT206	035776	EMT270	037462
EMS362	050202	EMS53	043317	EMT125	034426	EMT207	036026	EMT271	037506
EMS363	050223	EMS54	043360	EMT126	034446	EMT21	032414	EMT272	037532
EMS364	050240	EMS55	043435	EMT127	034464	EMT210	036042	EMT273	037550

R7 =X000007
 SADMSK= 000377
 SAVREG= 104414
 SA1 = 000001
 SA16 = 000020
 SA2 = 000002
 SA4 = 000004
 SA8 = 000010
 SC = 100000
 SCOPE = 000004
 SCTCMP 024302
 SCTMSK= 003700
 SCO = 000100
 SC1 = 000200
 SC2 = 000400
 SC3 = 001000
 SC4 = 002000
 SEARCH= 000030
 SEEK = 000004
 SETLFS 024410
 SETOM 023232
 SETVV 023110
 SHUT 023362
 SIZCLK 022500
 SKI = 040000
 SNGPRT= 020024
 STACK = 001100
 STANDA 004164
 START 002716
 STKLMT= 177774
 STOP 001534
 SWR 001154
 SWREG 000176
 SW0 = 000001
 SW00 = 000001
 SW01 = 000002
 SW02 = 000004
 SW03 = 000010
 SW04 = 000020
 SW05 = 000040
 SW06 = 000100
 SW07 = 000200
 SW08 = 000400
 SW09 = 001000
 SW1 = 000002
 SW10 = 002000
 SW11 = 004000
 SW12 = 010000
 SW13 = 020000
 SW14 = 040000
 SW15 = 100000
 SW2 = 000004
 SW3 = 000010
 SW4 = 000020
 SW5 = 000040
 SW6 = 000100
 SW7 = 000200

SW8 = 000400
 SW9 = 001000
 TADMSK= 177400
 TAG = 020000
 TAGADR= 001114
 TAP = 040000
 TA1 = 000400
 TA16 = 010000
 TA2 = 001000
 TA4 = 002000
 TA8 = 004000
 TBITVE= 000014
 TIME 001526
 TKVEC = 000060
 TPVEC = 000064
 TRAPVE= 000034
 TRE = 040000
 TRTVEC= 000014
 TST = 010000
 TSTNMB 022406
 TSTQUE 001462
 TST1 005376
 TST10 010662
 TST11 010746
 TST12 011216
 TST13 011546
 TST14 012132
 TST15 012460
 TST16 013130
 TST17 013554
 TST2 005706
 TST20 013776
 TST21 014470
 TST22 015456
 TST23 016724
 TST24 020114
 TST3 006070
 TST4 006240
 TST5 006370
 TST6 007454
 TST7 010524
 TYPBN = 104406
 TYPDS = 104405
 TYPE = 104401
 TYPOC = 104402
 TYPON = 104404
 TYPOS = 104403
 UBUSQS 031143
 UNS = 040000
 UNTMSK= 000007
 UPE = 020000
 USE = 040000
 U0 = 000001
 U1 = 000002
 U2 = 000004
 VV = 000100
 WATCH 001530

WC = 000040
 WCD = 000050
 WCE = 040000
 WCEHI = 010000
 WCELO = 004000
 WCF = 000040
 WCH = 000052
 WD = 000060
 WH = 000062
 WLE = 004000
 WRL = 004000
 XNUDC = 176000
 XNUER2= 001567
 XNUOF = 161577
 XSIZ 003756
 XXDP 001326
 \$APTHD 001100
 \$ATYC 030652
 \$ATY1 030626
 \$ATY3 030634
 \$ATY4 030644
 \$AUTOB 001150
 \$BASE 001276
 \$BDADR 001136
 \$BDDAT 001142
 \$BELL 001212
 \$BIN 025210
 \$CDW1 001302
 \$CDW2 001304
 \$CHARC 026214
 \$CKSWR 027300
 \$CMTAG 001114
 \$CM3 = 000000
 \$CM4 = 000005
 \$CNTLC 030176
 \$CNTLG 030210
 \$CNTLU 030203
 \$CPUOP 001250
 \$CRLF 001217
 \$DBLK 025426
 \$DDW0 001306
 \$DDW1 001310
 \$DDW2 001312
 \$DDW3 001314
 \$DDW4 001316
 \$DDW5 001320
 \$DDW6 001322
 \$DDW7 001324
 \$DEVCT 001232
 \$DEVVM 001300
 \$DOAGN 021556
 \$DTBL 025416
 \$ENDAD 021546
 \$ENDCT 021412
 \$ENULL 021562
 \$ENV 001242
 \$ENVM 001243

\$EOP 021356
 \$EOPCT 021404
 \$EOSP 021320
 \$ERFLG 001117
 \$ERMAX 001131
 \$ERROR 026602
 \$ERRPC 001132
 \$ERRTB 001536
 \$ERTTL 001126
 \$ESCAP 001210
 \$ETABL 001242
 \$ETEND 001326
 \$FATAL 001224
 \$FFLG 031072
 \$FILLC 001172
 \$FILLS 001171
 \$GDADR 001134
 \$GDDAT 001140
 \$GET42 021536
 \$GTSWR 027370
 \$HD = 000000
 \$HIBTS 001100
 \$HI OCT 030340
 \$ICNT 001120
 \$ILLUP 030610
 \$INTAG 001151
 \$ITEMB 001130
 \$LF 001220
 \$LFLG 031071
 \$LLCSR 001516
 \$LLVEC 001520
 \$LPADR 001122
 \$LP CSB 001510
 \$LP CSR 001506
 \$LPERR 001124
 \$LPVEC 001512
 \$MADR1 001254
 \$MADR2 001260
 \$MADR3 001264
 \$MADR4 001270
 \$MAIL 001222
 \$MAMS1 001252
 \$MAMS2 001256
 \$MAMS3 001262
 \$MAMS4 001266
 \$MBADR 001102
 \$MFLG 031070
 \$MNEW 030226
 \$MSGAD 001236
 \$MSGLG 001240
 \$MSGTY 001222
 \$MSWR 030215
 \$MTYP1 001253
 \$MTYP2 001257
 \$MTYP3 001263
 \$MTYP4 001267
 \$MXCNT 026530

\$NULL 001170
 \$NWTST= 000001
 \$OCNT 025660
 \$OMODE 025662
 \$OVER 026514
 \$PASS 001230
 \$PASTM 001106
 \$POWER 030616
 \$PSW 001524
 \$PWRDN 030450
 \$PW RMG 030604
 \$PW RUP 030522
 \$QUES 001216
 \$RDCHR 027642
 \$RD LIN 027732
 \$RDOCT 030240
 \$RDSZ = 000010
 \$RESRE 025100
 \$RM02 022460
 \$RM03 022465
 \$RM05 022472
 \$RTNAD 021560
 \$SAVRE 025042
 \$SAVR6 030614
 \$SCOPE 026220
 \$SETUP= 000137
 \$STUP = 177777
 \$SVLAD 026460
 \$SVPC = 000204
 \$SWR = 167400
 \$SWREG 001244
 \$SWRPMK= 000000
 \$SW08T 026532
 \$TESTN 001226
 \$TIMES 001206
 \$TKB 001162
 \$TKCNT 027002
 \$TKINT 027012
 \$TKQEN= 027011
 \$TKQIN 027004
 \$TKQOU 027006
 \$TKQSR 027010
 \$TKS 001160
 \$TKSRV 027062
 \$TMP0 001174
 \$TMP1 001176
 \$TMP2 001200
 \$TMP3 001202
 \$TMP4 001204
 \$TN = 000025
 \$TPB 001166
 \$TPFLG 001173
 \$TPS 001164
 \$TRAP 030342
 \$TRAP2 030402
 \$TRP = 000016
 \$TRPAD 030414

\$STM	001104	\$TYPE	025664	\$TYPOS	025436	\$VECT2	001274	\$\$GET4=	000000
\$STNM	001116	\$TYPEC	026076	\$UNIT	001234	\$XOFF =	000023	\$\$SW08=	000025
\$TYIN	030166	\$TYPEX	026216	\$UNITM	001110	\$XON =	000021	\$OFILL	025661
\$TYPB	025136	\$TYPOC	025462	\$USWR	001246	\$XTSTR	026232	.\$X	= 001100
\$TYPDS	025212	\$TYPON	025476	\$VECT1	001272				

. ABS. 056254 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 53600 WORDS (210 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 69 PAGES
CZRMQA.BIC,CZRMQA/C=CZRMQA.DOC,CZRMQA,SYSMAC/M

SMAIL 4-991 4-991 5-0# 8-19 8-25 10-27 12-1 M 12 12-73 12-106 12-137 12-159 12-326 12-518 12-545
SEQ 0155

STYPER 23-5 23-5 23-5

D 13

SEQ 0159

AMSGTV 5-0 5-0

F 13

SEQ 0161

BIT7	4-688#	8-68													
BIT8	4-688#														
BIT9	4-688#														
BOTADR	14-76*	14-94*	14-97	14-112	14-157#										
BOTFLG	14-61*	14-104*	14-107	14-110*	14-158#										
BPTVEC	4-688#														
BSE	4-881#														
BUFFER	12-562	12-633	12-688	12-814	12-925	12-:03	12-:45	12-:63	12-=:37	12-?67	12-?69	12-?84	33-5	33-5	
	33-21#	33-24													
BUF ONE	12-819	12-C58	33-22#												
BUF TWO	33-23#														
CC	4-873#														
CH	4-874#														
CHRCNT	14-62*	14-81*	14-87*	14-88	14-91*	14-95	14-100*	14-111*	14-159#						
CKSWR	23-6	23-7	23-7	23-10#											
CLKSNC	12-206	12-833	12-880	22-6#											
CLOCK	6-0#	12-571	12-590	15-13*	15-27*	15-39*	19-88	20-14							
CLR	4-946#	8-82	10-47	18-10											
CMNSTA	8-111	9-18	10-2#												
CNSL00	9-11	24-10#													
CNSL01	9-55	24-11#													
CNSL02	9-63	24-12#													
CNSL03	9-67	24-13#													
CNSL04	9-79	24-14#													
CNSL05	9-83	24-15#													
CNSL06	9-100	24-16#													
CNSL07	9-113	24-17#													
CNSL08	9-22	9-35	9-50	9-144	24-20#										
CNSL09	24-21#														
CNTCLR	12-7	12-28	12-75	12-108	12-116	12-139	12-162	12-213	12-262	12-365	12-381	12-403	12-425	12-442	
	12-464	12-480	12-495	12-547	12-610	12-679	12-750	16-14	18-8#	19-19					
COMMA	9-138	24-6#													
CONT	4-834#														
CR	4-688#	9-125	9-136	14-79	23-5	23-5	31-118	31-178	31-179	31-180	31-181	31-182	31-183	31-184	
	31-185	31-186	31-187	31-188	31-189	31-189	31-190								
CRLF	4-688#	8-6	8-25	8-25	8-39	8-50	8-56	8-57	10-24	18-17	18-17	23-5	23-5	24-4	
	24-7	24-9	24-10	24-11	24-12	24-14	24-16	24-17	24-18	24-19	24-20	24-21	24-22	32-1	
	32-3	32-5	32-9	32-11	32-14	32-16	32-18	32-20	32-22	32-24	32-26	32-28	32-30	32-32	
	33-27	33-28	33-29	33-30	33-31	33-32	33-33	33-34	33-35	33-36	33-37	33-38	33-39	33-40	
	33-41	33-42	33-43	33-44	33-45	33-46	33-47	33-48	33-49	33-50	33-51	33-52	33-53	33-54	
	33-55	33-56	33-57	33-58	33-59	33-60	33-61	33-62	33-63	33-64	33-65	33-66	33-67	33-68	
	33-69	33-70	33-71												
CYLMSK	4-863#	12-80													
DBCK	4-808#	19-106	19-119												
DBEN	4-809#	4-841	12-589	12-643	12-644	12-658	12-659	12-660	12-663	12-664	12-698	12-699	12-700	12-701	
	12-717	12-720	12-721	12-722	12-725	12-726	12-831	12-832	12-858	12-859	12-874	12-875	12-892	12-893	
	12-903	12-904	19-82	19-106	19-107	19-113	19-114	19-119	19-120	20-10	20-11	20-21	20-22	20-25	
	20-26	20-27	20-28	21-10	21-11	21-18	21-19								
DBL	4-964#														
DCK	4-770#	4-787	12-C46												
DDISP	4-688#	5-0	8-19												
DEBL	4-810#														
DISPLA	5-0#	8-19*	8-19*	23-6*	23-7*										
DISPRE	4-986#	8-19													
DLT	4-936#														
DMD	4-822#	4-840#	4-841	12-144	12-154	12-589	12-620	12-621	12-643	12-644	12-658	12-659	12-660	12-663	

12-664 12-681 12-682 12-698 12-699 12-700 12-701^{J 13} 12-717 12-720 12-721 12-722 12-725 12-726 12-755
SEQ 0165

EFS

30-3

30-11

33-16#

L 13

SEQ 0167

EMS100 27-50 27-54 27-56 27-62 27-64 27-67 27-69^{N 13} 27-104 31-68#

SEQ 0169

EMS313 27-15 27-132 27-138 27-145 27-160 27-167 27-174^{C 14} 27-184 31-105#

SEQ 0171

EMS314	27-17	27-21	27-25	27-29	27-47	31-106#										
EMS315	27-19	27-23	27-27	27-31	27-150	27-181	31-107#									
EMS316	27-17	27-19	27-21	27-23	27-25	27-27	27-29	27-31	31-108#							
EMS317	27-17	27-19	31-109#													
EMS32	27-189	31-27#														
EMS320	27-21	27-23	31-110#													
EMS321	27-25	27-27	31-111#													
EMS322	27-29	27-31	31-112#													
EMS323	31-113#															
EMS324	31-114#															
EMS325	31-115#															
EMS326	31-116#															
EMS327	27-45	27-72	27-76	27-94	27-117	27-211	27-246	27-261	27-261	27-282	27-283	27-293	31-117#			
EMS33	27-191	31-28#														
EMS330	27-47	31-118#														
EMS331	27-90	27-179	27-195	31-119#												
EMS332	27-98	27-140	27-147	27-162	27-169	27-176	27-186	27-324	27-353	27-369	27-387	27-404	31-120#			
EMS333	27-100	27-116	27-119	27-152	27-158	27-237	27-248	27-251	27-298	27-304	27-306	27-308	27-314	31-121#		
EMS334	27-62	27-121	27-152	27-156	27-242	27-293	27-298	31-122#								
EMS335	27-33	27-43	27-62	27-64	27-119	27-242	27-244	27-244	27-251	27-295	27-343	27-352	27-359	27-368		
	27-375	27-384	31-123#													
EMS336	27-35	27-60	27-67	27-69	27-112	27-121	27-339	27-344	27-355	27-360	27-371	27-376	27-391	27-395		
	27-406	27-410	31-124#													
EMS337	27-48	27-50	27-130	27-136	27-143	27-154	27-156	27-165	27-172	27-182	27-207	27-211	27-221	27-240		
	27-242	27-244	27-246	27-253	27-270	27-282	27-291	27-296	31-125#							
EMS34	27-193	27-255	31-29#													
EMS340	27-33	27-35	27-43	27-62	27-64	27-67	27-69	27-119	27-121	27-251	27-339	27-344	27-355	27-360		
	27-371	27-376	27-391	27-406	31-126#											
EMS341	27-50	27-104	27-134	27-141	27-143	27-148	27-154	27-156	27-163	27-165	27-170	27-172	27-182	27-191		
	27-193	27-203	27-207	27-211	27-262	27-264	27-266	27-268	27-270	27-272	27-274	27-278	27-310	31-127#		
EMS342	27-112	27-119	27-121	27-125	27-324	31-128#										
EMS343	27-86	27-125	31-129#													
EMS344	27-125	31-130#														
EMS345	27-128	27-130	27-205	27-276	27-293	31-131#										
EMS346	27-104	27-128	27-134	27-141	27-148	27-163	27-170	27-205	27-272	27-274	27-276	27-278	27-310	27-312		
	31-132#															
EMS347	27-132	27-138	27-145	27-160	27-167	27-174	27-184	31-133#								
EMS35	27-197	27-237	27-242	27-246	27-253	31-30#										
EMS350	27-150	27-152	27-156	27-158	27-298	27-314	27-332	27-337	31-134#							
EMS351	27-150	31-135#														
EMS352	27-37	27-39	27-158	31-136#												
EMS353	27-108	27-110	27-179	27-219	27-225	27-227	27-231	27-235	27-240	27-259	27-326	31-137#				
EMS354	27-187	27-189	27-213	27-219	27-223	27-231	27-235	27-259	27-280	27-285	27-287	27-289	27-328	31-138#		
EMS355	27-191	27-193	27-203	31-139#												
EMS356	27-41	27-318	27-320	27-322	27-389	31-140#										
EMS357	27-197	27-316	31-141#													
EMS36	27-201	27-215	27-221	27-223	27-225	27-227	27-251	27-316	27-343	27-348	27-352	27-359	27-364	27-368		
	27-375	27-380	27-384	27-400	31-31#											
EMS360	27-197	27-318	31-142#													
EMS361	27-320	31-143#														
EMS362	27-199	31-144#														
EMS363	27-209	27-300	27-302	27-314	31-145#											
EMS364	27-213	27-287	27-289	31-146#												
EMS365	27-64	27-213	27-219	27-225	27-227	27-231	27-235	27-259	27-324	27-326	27-328	31-147#				
EMS366	27-217	31-148#														
EMS367	27-219	27-225	27-227	27-231	27-235	27-259	27-326	31-149#								

EMS37 27-203 31-32#

E 14

SEQ 0173

EMS370	27-54	27-72	27-76	27-94	27-225	27-348	27-364	27-380	27-396	31-150#				
EMS371	27-106	27-227	27-398	31-151#										
EMS372	27-33	27-35	27-242	27-244	27-248	27-316	27-328	27-332	27-337	31-152#				
EMS373	27-276	27-293	31-153#											
EMS374	27-282	31-154#												
EMS375	27-304	31-155#												
EMS376	27-306	27-308	27-312	27-332	31-156#									
EMS377	27-33	27-41	27-64	27-324	27-328	27-332	27-341	27-350	27-357	27-366	27-373	27-382	27-393	27-408
	31-157#													
EMS4	27-90	27-195	31-4#											
EMS40	27-205	27-207	31-33#											
EMS400	27-35	27-70	27-337	31-158#										
EMS401	27-72	27-76	27-80	27-94	27-96	27-326	27-333	27-337	27-346	27-362	27-378	27-385	27-402	27-411
	31-159#													
EMS402	27-37	27-39	27-41	27-69	27-341	27-350	27-357	27-366	27-373	27-382	27-393	27-408	31-160#	
EMS403	27-341	27-350	27-357	27-366	27-373	27-382	27-393	27-408	31-161#					
EMS404	27-344	27-360	27-376	27-400	27-406	31-162#								
EMS405	27-84	27-92	27-339	27-344	27-346	27-348	27-353	27-355	27-360	27-362	27-364	27-369	27-371	27-376
	27-378	27-380	27-385	27-387	27-391	27-396	27-398	27-400	27-402	27-404	27-406	27-411	31-163#	
EMS406	27-37	27-39	27-339	27-355	27-371	27-391	31-164#							
EMS407	27-389	31-165#												
EMS41	27-209	27-211	27-213	27-242	27-244	27-248	27-322	27-324	31-34#					
EMS410	27-48	31-166#												
EMS411	27-43	27-48	31-167#											
EMS412	27-56	27-58	31-168#											
EMS413	27-65	27-82	27-84	27-92	27-102	27-108	31-169#							
EMS414	27-74	27-88	31-170#											
EMS415	27-65	27-67	27-69	31-171#										
EMS416	27-67	27-88	31-172#											
EMS417	27-74	27-80	27-82	27-96	27-102	31-173#								
EMS42	27-217	27-219	31-35#											
EMS420	27-82	27-102	31-174#											
EMS421	27-86	27-88	31-175#											
EMS422	27-110	31-176#												
EMS43	27-223	31-36#												
EMS44	27-229	27-231	31-37#											
EMS45	27-233	27-235	31-38#											
EMS46	27-238	27-240	31-39#											
EMS47	27-244	27-246	27-249	27-259	31-40#									
EMS5	27-100	27-300	31-5#											
EMS50	27-251	27-255	31-41#											
EMS500	27-5	31-178#												
EMS501	27-5	27-7	27-9	27-11	27-14	27-16	27-18	27-20	27-22	27-24	27-26	27-28	27-30	27-32
	27-46	27-53	27-79	27-91	27-99	27-101	27-113	27-115	27-118	27-120	27-122	27-124	27-135	27-142
	27-144	27-149	27-151	27-153	27-155	27-157	27-159	27-164	27-166	27-171	27-173	27-198	27-202	27-210
	27-212	27-214	27-216	27-222	27-224	27-236	27-241	27-243	27-245	27-247	27-250	27-252	27-254	27-256
	27-258	27-260	27-263	27-265	27-271	27-273	27-275	27-277	27-279	27-284	27-286	27-294	27-297	27-299
	27-301	27-303	27-305	27-307	27-309	27-311	27-313	27-315	27-317	27-319	27-321	27-323	27-325	27-327
	27-329	27-334	27-336	27-338	31-181#									
EMS502	27-5	27-7	27-9	27-11	27-14	27-16	27-18	27-20	27-22	27-24	27-26	27-28	27-30	27-32
	27-46	27-120	27-122	27-124	27-281	27-331	27-334	27-403	31-182#	27-24	27-26	27-28	27-30	27-32
EMS503	27-5	27-9	27-11	27-14	27-16	27-34	27-36	27-38	27-40	27-42	27-44	27-46	27-49	27-51
	27-53	27-59	27-61	27-73	27-77	27-79	27-81	27-85	27-93	27-95	27-97	27-107	27-127	27-129
	27-131	27-133	27-135	27-137	27-139	27-142	27-144	27-146	27-149	27-155	27-161	27-164	27-166	27-168
	27-171	27-173	27-175	27-178	27-180	27-183	27-185	27-188	27-190	27-200	27-216	27-218	27-220	27-226
	27-228	27-230	27-232	27-234	27-236	27-239	27-241	27-250	27-252	27-281	27-313	27-327	27-329	27-331

27-334 27-336 27-338 27-340 27-342 27-345 27-347^{G 14} 27-349 27-351 27-354 27-356 27-358 27-361 27-363
SEQ 0175

EMT1

7-3

27-3#

I 14

SEQ 0177

EMT10	7-25	27-17#
EMT100	27-132#	
EMT101	27-134#	
EMT102	7-199	27-136#
EMT103	27-138#	
EMT104	27-141#	
EMT105	27-143#	
EMT106	27-145#	
EMT107	27-148#	
EMT11	7-28	27-19#
EMT110	27-150#	
EMT111	27-152#	
EMT112	27-154#	
EMT113	27-156#	
EMT114	27-158#	
EMT115	27-160#	
EMT116	27-163#	
EMT117	27-165#	
EMT12	7-31	27-21#
EMT120	27-167#	
EMT121	27-170#	
EMT122	27-172#	
EMT123	27-174#	
EMT124	27-177#	
EMT125	27-179#	
EMT126	27-182#	
EMT127	27-184#	
EMT13	7-34	27-23#
EMT130	27-187#	
EMT131	27-189#	
EMT132	27-191#	
EMT133	27-193#	
EMT134	27-195#	
EMT135	27-197#	
EMT136	27-199#	
EMT137	27-201#	
EMT14	7-37	27-25#
EMT140	27-203#	
EMT141	27-205#	
EMT142	27-207#	
EMT143	27-209#	
EMT144	27-211#	
EMT145	27-213#	
EMT146	27-215#	
EMT147	27-217#	
EMT15	7-40	27-27#
EMT150	27-219#	
EMT151	27-221#	
EMT152	27-223#	
EMT153	27-225#	
EMT154	27-227#	
EMT155	27-229#	
EMT156	27-231#	
EMT157	27-233#	
EMT16	7-43	27-29#
EMT160	27-235#	

EMT161 27-238A

K 14

SEQ 0179

EMT162	27-240#	
EMT163	27-242#	
EMT164	27-244#	
EMT165	27-246#	
EMT166	27-249#	
EMT167	27-251#	
EMT17	7-46	27-31#
EMT170	7-193	27-253#
EMT171	27-255#	
EMT172	27-257#	
EMT173	7-202	27-259#
EMT174	7-205	27-261#
EMT175	7-208	27-264#
EMT176	7-211	27-266#
EMT177	7-214	27-268#
EMT2	7-6	27-4#
EMT20	7-49	27-33#
EMT200	7-217	27-270#
EMT201	27-272#	
EMT202	27-274#	
EMT203	27-276#	
EMT204	27-278#	
EMT205	27-280#	
EMT206	27-282#	
EMT207	27-285#	
EMT21	7-52	27-35#
EMT210	27-287#	
EMT211	27-289#	
EMT212	27-291#	
EMT213	27-293#	
EMT214	27-296#	
EMT215	27-298#	
EMT216	27-300#	
EMT217	27-302#	
EMT22	7-55	27-37#
EMT220	27-304#	
EMT221	27-306#	
EMT222	27-308#	
EMT223	27-310#	
EMT224	27-312#	
EMT225	27-314#	
EMT226	27-316#	
EMT227	27-318#	
EMT23	7-58	27-39#
EMT230	27-320#	
EMT231	27-322#	
EMT232	27-324#	
EMT233	27-326#	
EMT234	27-328#	
EMT235	27-330#	
EMT236	27-332#	
EMT237	27-335#	
EMT24	7-61	27-41#
EMT240	27-337#	
EMT241	27-339#	
EMT242	27-341#	

EMT243 27-3440

M 14

SEQ 0181

EMT244	27-346#		
EMT245	27-348#		
EMT246	27-350#		
EMT247	27-353#		
EMT25	7-64	27-43#	
EMT250	27-355#		
EMT251	27-357#		
EMT252	27-360#		
EMT253	27-362#		
EMT254	27-364#		
EMT255	27-366#		
EMT256	27-369#		
EMT257	27-371#		
EMT26	7-67	27-45#	
EMT260	27-373#		
EMT261	27-376#		
EMT262	27-378#		
EMT263	27-380#		
EMT264	27-382#		
EMT265	27-385#		
EMT266	27-387#		
EMT267	27-389#		
EMT27	7-70	27-48#	
EMT270	27-391#		
EMT271	27-393#		
EMT272	27-396#		
EMT273	27-398#		
EMT274	27-400#		
EMT275	7-196	27-402#	
EMT276	7-130	27-404#	
EMT277	27-406#		
EMT3	7-9	27-6#	
EMT30	7-73	27-50#	
EMT300	27-408#		
EMT301	27-411#		
EMT31	7-76	27-52#	
EMT32	7-79	27-54#	
EMT33	7-82	27-56#	
EMT34	7-85	27-58#	
EMT35	7-88	27-60#	
EMT36	7-91	27-62#	
EMT37	7-94	27-64#	
EMT4	7-12	27-8#	
EMT40	7-97	27-67#	
EMT41	7-100	27-69#	
EMT42	7-103	27-72#	
EMT43	7-106	7-109	27-74#
EMT44	27-76#		
EMT45	7-112	27-78#	
EMT46	7-115	27-80#	
EMT47	7-118	27-82#	
EMT5	7-15	27-10#	
EMT50	7-121	27-84#	
EMT51	7-124	27-86#	
EMT52	7-127	27-88#	
EMT53	27-90#		

EMT54 7-133 27-92#

B 15

SEQ 0183

HELP

9-37

33-26#

D 15

SEQ 0185

LSTRK 6-0# 10-45* 10-53* 12-:60 12-:69 12-=13 12-=35^{F 15} 12->90 12->93 12-?81 12-B16

SEQ 0187

PCOUNT 15- 15-69#

H 15

SEQ 0189

PDA	4-832#	12-a25	12-a69	12-a79	12-a93	12-a94	12-a97	12-A17	12-C21	12-C25					
PGE	4-941#														
PGM	4-762#														
PHA	4-833#	12-;98	12-<04	12-<64	12-<68	12-a25	12-A62	12-A64							
PIP	4-758#														
PIRQ	4-688#														
PIRQVE	4-688#														
PLCLK	15-54	15-58#													
PLFS	4-830#	12-842	12-881	12-883	12->52	12->55	12-?30	12-?37	12-B44	12-B51	12-B64	12-B70	12-B73	21-21	
	22-10	22-13	22-14												
PLSTP	15-82	15-86#													
PRO	4-688#														
PR1	4-688#														
PR2	4-688#														
PR3	4-688#	8-23													
PR4	4-688#														
PR5	4-688#	15-64													
PR6	4-688#	8-22	10-40	15-9	15-16	15-30	15-62								
PR7	4-688#	15-61													
PS	4-688	4-688#													
PSEL	4-928#														
PSTOP	15-14	15-81#													
PSW	4-688#														
PUTBUF	6-0#														
PWRVEC	4-688#	8-19*	8-19*	23-11*	23-11*	23-11*	23-11*	23-11*	23-11*	23-11*					
QUES	9-57	9-73	9-94	24-5#											
R6	4-688#	8-19	8-19*	8-19*											
R7	4-688#														
RD	4-729#	12-927	12-:05	12-B22											
RDCHR	9-12	9-28	9-43	9-117	9-134	23-8	23-10#								
RDLIN	23-9	23-10#													
RDOCT	9-58	9-74	9-95	23-10#											
RDY	4-931#														
READY	10-29#	13-16	13-20												
RECAL	4-707#														
RESREG	14-152	23-10#													
RESVEC	4-688#														
REX	4-828#														
RG	4-826#	19-90	19-95												
RH	4-730#	12-=40													
RIP	4-712#														
RELEASE	4-709#														
RMAS	4-902#	12-58	12-385*												
RMASI	6-0#														
RMASO	6-0#														
RMBAA	4-975#	12-54	19-79*												
RMBAAE	4-978#														
RMBAAEI	6-0#														
RMBAAEO	6-0#														
RMBAAI	6-0#														
RMBAAO	6-0#	12-562*	12-633*	12-688*	12-814*	12-925*	12-:03*	12-:45*	12-:64*	12-=37*	12-?85*	12-B20*	19-79		
RMCS1	4-898#	4-973#	8-89	12-78*	12-85	12-111*	12-119	12-165*	12-172*	12-180	12-222*	12-230	12-428*	12-548	
	16-19*	17-17*	19-61*	19-85*											
RMCS1I	6-0#	12-85*	12-93*	12-119*	12-124*	12-126	12-180*	12-189*	12-193*	12-199	12-230*	12-238*	12-243		
RMCS1O	6-0#	12-564*	12-635*	12-690*	12-816*	12-927*	12-:05*	12-:47*	12-:72*	12-=40*	12-?88*	12-B22*	19-85		
RMCS2	4-976#	8-82*	8-83*	8-86	10-47*	10-48*	12-8	12-15	12-25	12-29	12-36	12-44	12-56	18-10*	

18-11•

J 15

SEQ 0191

RMCS21	6-0#													
RMCS20	6-0#													
RMCS3	4-979#													
RMCS3I	6-0#													
RMCS30	6-0#													
RMDA	4-899#	12-79*	12-86	12-166*	12-173*	12-181	12-217*	12-223*	12-231	12-265*	12-272	12-382*	12-388	12-404*
	12-410	12-465*	12-469	12-482*	12-486	12-501*	19-75*							
RMDAI	6-0#	12-86*	12-95	12-181*	12-194*	12-200	12-231*	12-244	12-272*	12-281	12-388*	12-391	12-410*	12-413
	12-469*	12-471	12-486*	12-489										
RMDAO	6-0#	12-559*	12-630*	12-685*	12-812*	12-922*	12-:00*	12-:42*	12-;60*	12-;61*	12-=35*	12-=36*	12-?81*	12-?82*
	12-B16*	12-B17*	19-75											
RMDB	4-977#	12-60												
RMDBI	6-0#													
RMDBO	6-0#													
RMDC	4-908#	12-80*	12-87	12-176*	12-184	12-219*	12-225*	12-233	12-267*	12-274	12-367*	12-371	12-427*	12-431
	12-445*	12-451	12-483*	12-497*	12-503	19-76*								
RMDCI	6-0#	12-87*	12-97*	12-184*	12-191*	12-197*	12-203	12-233*	12-240*	12-246	12-274*	12-291	12-371*	12-374*
	12-375	12-431*	12-435*	12-436	12-451*	12-457*	12-458	12-503*	12-507	12-508				
RMDCO	6-0#	12-560*	12-631*	12-686*	12-811*	12-923*	12-:01*	12-:43*	12-;59*	12-=34*	12-?80*	12-B15*	19-76	
RMDS	4-900#	8-84	12-467*	12-499*	16-20	16-24	17-18	17-23	19-62	19-66				
RMDSI	6-0#													
RMDSO	6-0#													
RMDT	4-905#	10-49	12-446*	12-520	12-540	14-27								
RMDTI	6-0#													
RMDTO	6-0#													
RMEC1	4-912#	12-448*												
RMEC1I	6-0#													
RMEC10	6-0#													
RMEC2	4-913#	12-62	12-409*											
RMEC2I	6-0#													
RMEC20	6-0#													
RMER1	4-901#	12-112*	12-120	12-142*	12-147	12-167*	12-174*	12-182	12-226*	12-234	12-268*	12-275	12-366*	12-370
	12-426*	12-430	12-443*	12-449	12-481*	12-485	12-500*	12-577	12-579	12-596	12-606	12-612	12-622	12-650
	12-665	12-704	12-708	12-727	12-731	12-?56	12-?61	12-C45	12-C52	16-17*	17-15*	19-59*	19-83*	
RMER1I	6-0#	12-120*	12-128	12-147*	12-150*	12-182*	12-195*	12-201	12-234*	12-247	12-275*	12-297	12-370*	12-372
	12-430*	12-432*	12-433	12-449*	12-452	12-485*	12-487							
RMER10	6-0#													
RMER2	4-911#	12-113*	12-121	12-143*	12-148	12-169*	12-177*	12-185	12-227*	12-235	12-269*	12-276	12-384*	12-390
	12-406*	12-412	12-447*	12-466*	12-470	12-484*	12-498*	12-504	16-18*	17-16*	19-60*	19-84*		
RMER2I	6-0#	12-121*	12-125*	12-130	12-148*	12-152*	12-185*	12-192*	12-198*	12-204	12-235*	12-241*	12-248	12-276*
	12-301	12-390*	12-396*	12-397	12-412*	12-418*	12-419	12-470*	12-473*	12-474	12-504*	12-510*	12-511	
RMER20	6-0#													
RMHR	4-909#	12-369*	12-408*											
RMHRI	6-0#													
RMHRO	6-0#	33-9												
RMLA	4-903#	12-407*												
RMLAI	6-0#													
RMLAO	6-0#													
RMMR1	4-904#	12-144*	12-149	12-368*	12-589*	12-620*	12-621*	12-643*	12-644*	12-658*	12-659*	12-660*	12-663*	12-664*
	12-681*	12-682*	12-698*	12-699*	12-700*	12-701*	12-717*	12-720*	12-721*	12-722*	12-725*	12-726*	12-752	12-755*
	12-757*	12-758*	12-759	12-768*	12-769*	12-770	12-782*	12-783*	12-784	12-796*	12-797*	12-798	12-831*	12-832*
	12-835	12-846	12-858*	12-859*	12-862	12-874*	12-875*	12-880	12-885	12-892*	12-893*	12-903*	12-904*	12-905
	12-911	12-943	12-945*	12-947*	12-948*	12-949	12-962*	12-963*	12-964	12-983*	12-985*	12-987*	12-988	12-:20*
	12-:25*	12-:26*	12-:27	12-:34	12-:86*	12-:87*	12-:88	12-:98*	12-:99*	12-:00	12-:11	12-:20*	12-:21*	12-:36
	12-:43*	12-:44*	12-:87*	12-:88*	12-:89	12-<07	12-<13*	12-<14*	12-<15	12-<38*	12-<39*	12-<50	12-<55*	12-<56*
	12-<67	12-<76	12-<79*	12-<80*	12-<85	12-<95	12-=00*	12-=01*	12-=57*	12-=58*	12-=59	12-=78*	12-=79*	12-=80

12->00* 12->01* 12->02 12->22* 12->23* 12->24 12->41*^{L 15} 12->42* 12->43 12->57 12->65* 12->66* 12->80* 12->82*
SEQ 0193

STACK 4-688# 8-19 12-1 12-73 12-106 12-137 12-159^{N 15} 12-326 12-518 12-545 12-556 12-603 12-673 12-748
SEQ 0195

TST14 12-748# 23-6

C 16

SEQ 0197

TST15	12-808#	23-6												
TST16	12-919#	23-6												
TST17	12-997#	23-6												
TST2	12-73#	23-6												
TST20	12-:39#	23-6												
TST21	12-:56#	23-6												
TST22	12-:31#	23-6												
TST23	12-?77#	23-6												
TST24	12-812#	23-6												
TST3	12-106#	23-6												
TST4	12-137#	23-6												
TST5	12-159#	23-6												
TST6	12-326#	23-6												
TST7	12-518#	23-6												
TSTNMB	14-45*	14-46*	14-48	14-155#										
TSTQUE	6-0#	10-4	10-5*	10-41	10-48	12-1	12-73	12-106	12-137	12-159	12-326	12-518	12-545	12-556
	12-603	12-673	12-748	12-808	12-919	12-997	12-:39	12-:56	12-:31	12-?77	12-812	13-11	13-13*	13-17
	13-17*	18-11												
TYPBN	14-145	23-10#												
TYPDS	13-20	13-20	14-139	23-10#										
TYPE	8-6	8-25	8-39	8-45	8-50	8-56	8-57	8-94	8-95	8-97	8-104	9-11	9-14	9-17
	9-22	9-27	9-30	9-35	9-37	9-41	9-42	9-45	9-50	9-55	9-57	9-63	9-67	9-73
	9-79	9-83	9-94	9-100	9-113	9-114	9-116	9-121	9-127	9-138	9-139	9-144	9-152	10-24
	13-20	13-20	13-20	14-22	14-23	14-40	14-41	14-47	14-52	14-54	14-60	14-99	14-105	14-109
	14-122	14-128	14-130	14-148	14-150	18-17	23-2	23-3	23-4	23-5	23-7	23-7	23-8	23-8
	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8	23-8
	23-8	23-8	23-10#	23-11										
TYPOC	8-8	9-56	14-142	23-8	23-10#									
TYPON	23-10#													
TYPOS	8-42	8-53	8-96	9-70	9-91	14-24	14-48	14-53	14-55	23-10#				
U0	4-951#													
U1	4-951#													
U2	4-951#													
UBUSQS	9-42	24-8#												
UNS	4-771#	12-150												
UNTMSK	4-955#	12-12	12-33											
UPE	4-938#													
USE	4-894#													
VV	4-765#	16-21	16-25	19-63	19-67									
WATCH	6-0#	12-571*	12-572	12-590*	12-591	15-74*	15-76*	19-88*	19-92	20-14*	20-15			
WC	4-835#	12-760	12-771	12-775	12-785	12-788	12-799	12-836	12-863	12-906	12-950	12-965	12-989	12-993
	12-:28	12-:32	12-:89	12-:01	12-:90	12-<16	12-=60	12-=81	12->03	12->25	12->44	12-?22	12-?48	12-A75
	12-A80	12-A94	12-C76	12-C81	12-C95	19-26	19-33	19-40	19-46	21-13	21-33	21-39	22-21	22-31
WCD	4-721#													
WCE	4-937#	25-77	25-78											
WCEHI	4-962#													
WCELO	4-963#													
WCF	4-780#	4-787												
WCH	4-722#													
WD	4-725#	12-564	12-635	12-690	12-816	12-?88								
WH	4-726#	12-:47	12-:72											
WLE	4-774#	4-787	25-81	25-82										
WRL	4-760#													
XNUDC	4-864#	12-97	12-191	12-225	12-240	12-292	12-295	12-374	12-435	12-457	12-507			
XNUER2	4-889#	12-125	12-192	12-227	12-241	12-302	12-305	12-396	12-418	12-473	12-510			
XNUOF	4-859#	12-81	12-99	12-190	12-224	12-239	12-286	12-289	12-393	12-415	12-454	12-505		

XSIZ	8-67#	9-123	9-153						
XXDP	6-0#	8-30*	8-33*	8-34	8-36*	8-41	8-52	8-77	8-79

GETAS	4-278#													
GETBA	4-142#													
GETBAE	4-286#													
GETCS1	4-126#	12-85	12-119	12-180	12-230	12-548								
GETCS2	4-158#	12-8	12-29											
GETDA	4-182#	12-86	12-181	12-231	12-272	12-388	12-410	12-469	12-486					
GETDB	4-150#													
GETDC	4-190#	12-87	12-184	12-233	12-274	12-371	12-431	12-451	12-503					
GETDS	4-166#	16-20	17-18	19-62										
GETDT	4-214#	12-520	14-27											
GETEC1	4-230#													
GETEC2	4-238#													
GETER1	4-174#	12-120	12-147	12-182	12-234	12-275	12-370	12-430	12-449	12-485	12-577	12-596	12-612	12-622
	12-650	12-665	12-704	12-727	12-756	12-C45								
GETER2	4-206#	12-121	12-148	12-185	12-235	12-276	12-390	12-412	12-470	12-504				
GETHR	4-246#													
GETLA	4-198#													
GETMR1	4-254#	12-149	12-759	12-770	12-784	12-798	12-835	12-862	12-880	12-905	12-949	12-964	12-988	12-:27
	12-:88	12-:00	12-:11	12-:36	12-:89	12-<15	12-<50	12-<76	12-<95	12-=59	12-=80	12->02	12->24	12->43
	12-?21	12-?47	12-@24	12-@68	12-@92	12-A16	12-A37	12-A74	12-A93	12-B43	12-B63	12-C20	12-C75	12-C94
	19-25	19-39	19-89	19-123	21-12	21-20	21-32	21-38	22-9	22-25				
GETMR2	4-262#	12-:59	12-:73	12-<24	12-=68	12->11	12-@38	12-C33						
GETOF	4-270#	12-88	12-183	12-232	12-273	12-389	12-411	12-450	12-502					
GETPRI	4-688#	15-59												
GETSN	4-222#													
GETSWR	4-688#	8-25	8-25#	10-27										
GETWC	4-134#													
GETX	4-118#													
MSG	4-8#	12-1	12-73	12-106	12-137	12-159	12-326	12-518	12-545	12-556	12-603	12-673	12-748	12-808
	12-919	12-997	12-:39	12-:56	12-=31	12-?77	12-B12							
MULT	4-688#													
NEWTST	4-688#	12-1	12-73	12-106	12-137	12-159	12-326	12-518	12-545	12-556	12-603	12-673	12-748	12-808
	12-919	12-997	12-:39	12-:56	12-=31	12-?77	12-B12							
NWTST	4-504#	12-1	12-73	12-106	12-137	12-159	12-326	12-518	12-545	12-556	12-603	12-673	12-748	12-808
	12-919	12-997	12-:39	12-:56	12-=31	12-?77	12-B12							
POP	4-688#	15-46	15-47	18-12	23-1	23-3	23-9	23-11	23-11	23-12	23-12			
PUSH	4-688#	15-6	15-7	18-8	23-1	23-3	23-9	23-11	23-11	23-12	23-12	23-12		
PUTAS	4-466#	12-385												
PUTBA	4-330#	19-79												
PUTBAE	4-474#													
PUTCS1	4-315#	12-78	12-111	12-165	12-172	12-222	12-428	16-19	17-17	19-61	19-85			
PUTCS2	4-346#													
PUTDA	4-370#	12-79	12-166	12-173	12-217	12-223	12-265	12-382	12-404	12-465	12-482	12-501	19-75	
PUTDB	4-338#													
PUTDC	4-378#	12-80	12-176	12-219	12-225	12-267	12-367	12-427	12-445	12-483	12-497	19-76		
PUTDS	4-354#	12-467	12-499											
PUTDT	4-402#	12-446												
PUTEC1	4-418#	12-448												
PUTEC2	4-426#	12-409												
PUTER1	4-362#	12-112	12-142	12-167	12-174	12-226	12-268	12-366	12-426	12-443	12-481	12-500	16-17	17-15
	19-59	19-83												
PUTER2	4-394#	12-113	12-143	12-169	12-177	12-227	12-269	12-384	12-406	12-447	12-466	12-484	12-498	16-18
	17-16	19-60	19-84											
PUTHR	4-434#	12-369	12-408											
PUTLA	4-386#	12-407												
PUTMR1	4-442#	12-144	12-368	12-589	12-620	12-621	12-643	12-644	12-658	12-659	12-660	12-663	12-664	12-681

12-682 12-698 12-699 12-700 12-701 12-717 12-720^{H 16} 12-721 12-722 12-725 12-726 12-755 12-757 12-758
SEQ 0202

	12-768	12-769	12-782	12-783	12-796	12-797	12-831	12-832	12-858	12-859	12-874	12-875	12-892	12-893
	12-903	12-904	12-945	12-947	12-948	12-962	12-963	12-983	12-985	12-987	12-:20	12-:25	12-:26	12-:86
	12-:87	12-:98	12-:99	12-:20	12-:21	12-:43	12-:44	12-:87	12-:88	12-<13	12-<14	12-<38	12-<39	12-<5>
	12-<56	12-<79	12-<80	12-=00	12-=01	12-=57	12-=58	12-=78	12-=79	12->00	12->01	12->22	12->23	12->41
	12->42	12->65	12->66	12->80	12->82	12-?05	12-?07	12-?19	12-?20	12-?45	12-?46	12-A21	12-A22	12-A43
	12-A44	12-A72	12-A73	12-A91	12-A92	12-B87	12-B88	12-C04	12-C06	12-C08	12-C73	12-C74	12-C92	12-C93
	16-15	16-16	17-14	19-20	19-23	19-24	19-37	19-38	19-56	19-57	19-82	19-106	19-107	19-113
	19-114	19-119	19-120	20-10	20-11	20-21	20-22	20-25	20-26	20-27	20-28	21-10	21-11	21-18
	21-19	21-35	21-36	21-41	21-42	22-23	22-24	22-49	22-51	22-53				
PUTMR2	4-450#	12-387												
PUTOF	4-458#	12-81	12-168	12-175	12-218	12-224	12-266	12-383	12-405	12-444	12-468	12-496	19-77	
PUTSN	4-410#	12-386	12-429											
PUTWC	4-323#	19-78												
PUTX	4-307#													
REPORT	4-688#													
RGBFMC	4-20#	6-0	6-0											
SCTCMP	4-638#	12-821	12-932	12-:10	12-:52	12-:77	12-=45	12-@00	12-B27					
SETLFS	4-649#	12-935	12-:13	12-@03	12-B30									
SETOM	4-615#													
SETPRI	4-688#	8-23	10-40	15-64	15-87	23-8								
SETTRA	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10	23-10#
SETUP	4-688#	8-19												
SETVV	4-604#													
SKIP	4-688#													
SLASH	4-688#													
STARS	4-688#	4-988	4-991	4-991	4-991	5-0	5-0	5-0	12-1	12-1	12-73	12-73	12-106	12-106
	12-137	12-137	12-159	12-159	12-326	12-326	12-518	12-518	12-545	12-545	12-556	12-556	12-603	12-603
	12-673	12-673	12-748	12-748	12-808	12-808	12-919	12-919	12-997	12-997	12-:39	12-:39	12-:56	12-:56
	12-=31	12-=31	12-?77	12-?77	12-B12	12-B12	13-20	14-2	23-1	23-2	23-3	23-4	23-5	23-6
	23-7	23-8	23-8	23-8	23-8	23-8	23-9	23-10	23-11	23-11	23-12			
SWRSU	4-688#	8-19	8-19#											
TAGS	4-54#	5-0												
TRMTRP	23-10#													
TYPBIN	4-688#													
TYFDEC	4-688#	13-20	13-20											
TYPNAM	4-676#	4-688#	8-25											
TYPNUM	4-688#													
TYPOCS	4-688#	8-96	14-24	14-48	14-53	14-55								
TYPOCT	4-688#	9-56	23-8											
TYPTXT	4-688#	8-6	8-39	8-50	8-56	8-57	10-24	13-20	13-20	18-17				